

PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

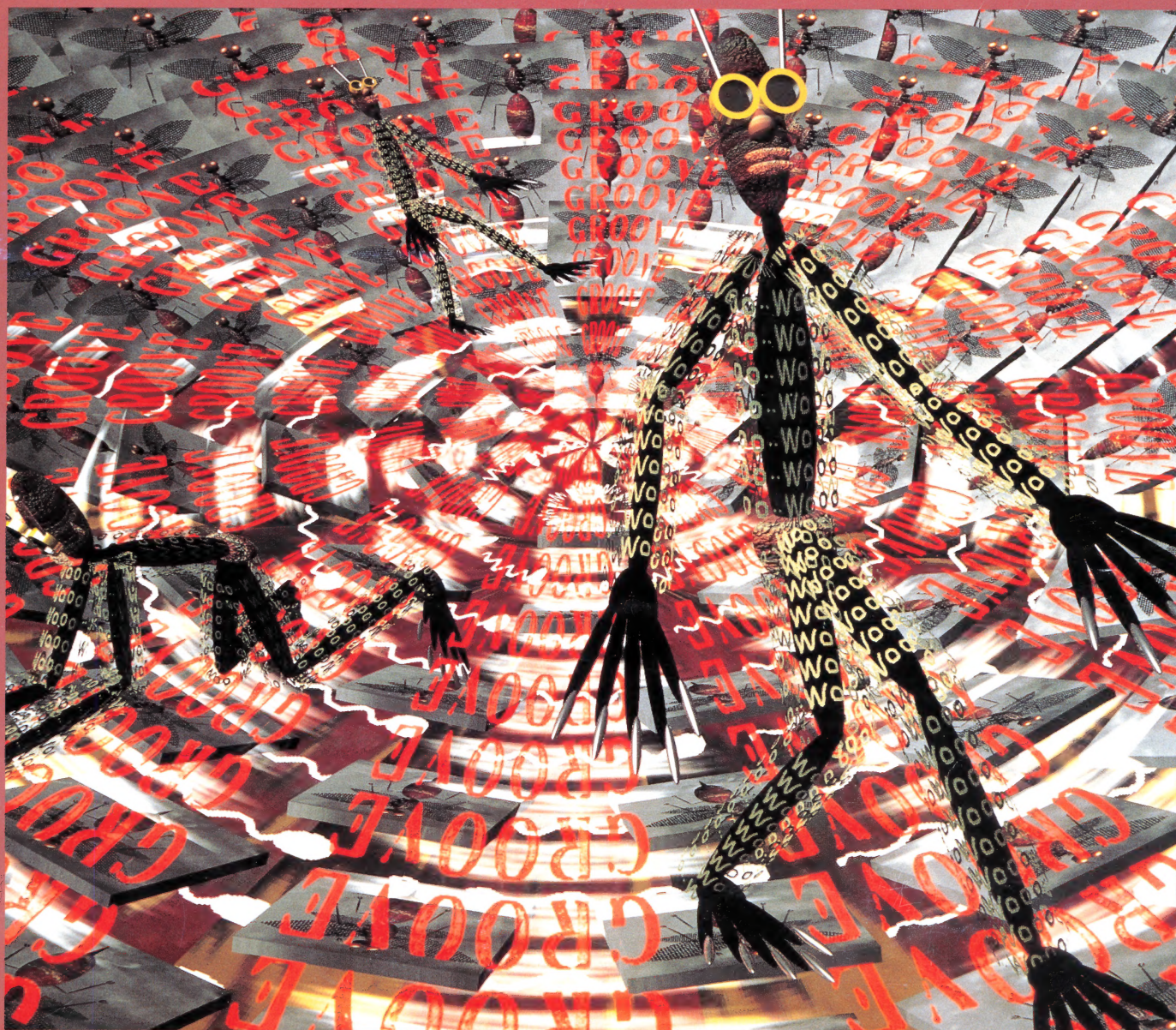
# PCX

## 特集 ポリゴナイザSLASHの活用

モデラの拡張/回転体生成プログラム/ポリゴンソート関数  
スクリーンセーバーのモジュールを作る/CASSAVE.X&CASLOAD.X  
新製品紹介 SX-PhotoGallery/ Easydraw SX-68K/FLICKER/OS-9

11

1993





# SHARP

夢  
の  
頂  
き  
へ。  
68  
ワ  
ル  
ド  
の  
最  
高  
峰。



目の付けどころが、

シャープでしょ。



 **68030**  
32bit PERSONAL WORKSTATION



## 演算速度4.3倍(当社10MHz機比)/2.4倍(当社XVI比)\*1、動画ウィンドウに見る新創造次元。 選ばれた人だけが持つ感性によってX68030の扉はひらかれる。

X68000シリーズとして初の32ビットMPU MC68EC030を搭載して高速化を実現。

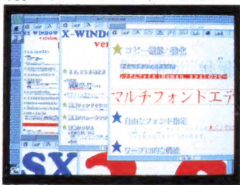
データキャッシュ、プログラムキャッシュをそれぞれ256バイト搭載したクロック周波数25MHzの高速32ビットMPUを搭載。演算速度は2倍以上(当社従来比)\*1の高速化を実現しました。また数値演算プロセッサMC68882\*2(25MHz)もサポート。大量の実数演算を必要とするクリエイティブワークやGUI環境の操作性など、実行速度の飛躍的な向上が図られています。(当社従来比)

\*1 Dhrystn(四則演算)比。25MHz・データキャッシュオン・プログラムキャッシュオンでMC68000/10MHz時の約4.3倍、16MHz時の約2.4倍。

\*2 数値演算プロセッサCZ-5MP1標準価格54,800円(税別)  
: 本体内の専用ソケットに取り付け可能。

65,536色表示、動画表示を実現。さらにパワーアップしたSX-WINDOW ver.3.0。

X68000独自のウィンドウシステムとして定評の「SX-WINDOW ver.2.0」をさらに強化した「SX-WINDOW ver.3.0」を標準装備。



新たに、65,536色の自然色グラフィック表示を可能とした『グラフィックウィンドウ』\*を搭載。またアニメーション動画をウィンドウ上で表現でき、手軽にコンピュータアニメーションが楽しめる『CGAウィンドウ』。さらに従来のエディタのイメージを一新、高度な日本語文書作成をサポートするSX-WINDOW対応の高機能日本語マルチフォントエディタを標準装備。アウトラインフォントの展開もさらに高速化が図られています。

\*SX-WINDOW上の512×512ドットのエリア内で表示可能。

GUIに対応する大容量メインメモリを搭載。

メインメモリは標準で4Mバイト、複数のアプリケーションをウィンドウ上で同時に使用するなど大量のデータ処理に

応。また本体内の増設で、I/Oスロットを使用せず最大12Mバイトまで拡張できます。拡張したメモリはすべて32ビットバスによる高速アクセスが可能、優れた拡張環境でシステムパワーアップをサポートします。

\*メモリ増設には、4MB内部増設RAMボードCZ-5BE4標準価格54,800円(税別)、4MB増設RAMモジュールCZ-5ME4標準価格49,800円(税別)をご使用ください。なおCZ-5ME4はCZ-5BE4上に装着します。

X68000シリーズの高機能を継承した上で、さらに使いやすさの向上を図ったコンパチビリティ重視設計\*1、すぐに使える高機能ソフトを標準装備。

●25MHzでは速すぎるアプリケーションも、従来のクロック周波数(10MHz/16MHz)で動作可能なソフトコンパチ重視設計●65,536色同時発色の自然色グラフィックス(最大表示エリア512×512ドット)、1024×1024ドットの実画面エリアを持つ高解像度表示能力(最大表示エリア768×512ドット・カラー液晶ディスプレイ使用時\*2は640×480ドット)、疑似高解像度スーパーインポーズ(インターレース方式/512×480ドット・専用ディスプレイテレビ使用時)を装備した高精細度自然色グラフィックス機能。●外部MIDI音源もコントロール可能\*3、ウィンドウ上で手軽にコンピュータミュージックが楽しめるMIDI音源対応デバイスドライバ搭載●ステレオ8オクターブ8重和音FM音源、ADPCM搭載●プリンタ、RS-232C、SCSI、オーディオ入出力、イメージス力など多彩なインターフェイスを装備。●日本語変換効率や操作性を高めた日本語フロントプロセッサASK68K ver.3.0搭載。●従来のエディタのイメージを一新したSX-WINDOW対応の高速多機能日本語マルチフォントエディタ標準装備●日本語マルチフォントエディタ中に貼り付ける絵やグラフなどが簡単に作成できるグラフィックパターナーエディタ●MIDI対応のX-BASIC。

\*1 アプリケーションソフトおよび周辺機器のうち、一部動作しないものがあります。詳しくはシャープお客様相談窓口にお問い合わせください。

\*2 10.4型カラー液晶ディスプレイLC-10C1-H標準価格598,000円(税別)、接続ケーブルAN-1515X標準価格4,200円(税別)をご使用ください。(SX-WINDOW対応アプリケーションのみ。色数に制限があります。)

\*3 別売のMIDIインターフェイスが必要。

### 5.25" FDDマンハッタンシェイプシリーズ



■X68000伝統のマンハッタンシェイプを継承 ■5.25インチFDD2基搭載

■80MBハードディスク内蔵(CZ-510C)\*

■マウス・トラックボール標準装備 ■ASCII準拠フルキーボード採用

\*CZ-500Cには、2.5インチ80MB内蔵用ハードディスクドライブCZ-5H08/2.5インチ160MB内蔵用ハードディスクドライブCZ-5H16を用意しています。

**X68030**  
32bit PERSONAL WORKSTATION

本体+キーボード+マウス・トラックボール  
5.25インチFDDタイプ CZ-500C-B(チタンブラック)標準価格398,000円(税別)  
■HDタイプ CZ-510C-B(チタンブラック)標準価格488,000円(税別)  
14型カラーディスプレイ  
CZ-608D-B(チタンブラック)標準価格94,800円(税別・チルトスタンド同梱)

### 3.5" FDDコンパクトシリーズ

■32ビットのハイパワーを凝縮したコンパクトフォーム ■2DD対応3.5インチFDD2基搭載  
■80MBハードディスク内蔵(CZ-310C)\* ■マウス標準装備 ■コンパクトキーボード採用  
\*CZ-300Cには、2.5インチ80MB内蔵用ハードディスクドライブCZ-5H08/2.5インチ160MB内蔵用ハードディスクドライブCZ-5H16を用意しています。

**X68030**  
32bit PERSONAL WORKSTATION  
Compact

本体+キーボード+マウス

3.5インチFDDタイプ CZ-300C-B(チタンブラック)標準価格388,000円(税別)

■HDタイプ CZ-310C-B(チタンブラック)標準価格478,000円(税別)

14型カラーディスプレイ

CZ-608D-B(チタンブラック)標準価格94,800円(税別・チルトスタンド同梱)



**X68030**  
32bit PERSONAL WORKSTATION  
&  
**X68000**  
PERSONAL WORKSTATION・XVI

68買ったら  
EXEクラブへ  
入ろう!

EXE  
クラブって  
何だ?

X68030/X68000を手に入れたら、  
やっぱり他のユーザーがどんな  
風に使っているのか気になるもの。  
ということでEXEクラブは、そん  
なあなたのための、他の68ユー  
ザーとのコミュニケーションをバツ  
クアップする、情報交換の場です。

本体同梱の入会申込ハガキを  
送るだけで、自動的に無料入会。  
さらに下記の特典付き。

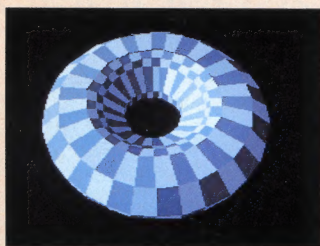
メリット  
1

会員ナンバー入りオリジナル  
会員電卓がもらえる。

メリット  
2

各種フェアご優待・イベント  
案内等、数々の特典がある。

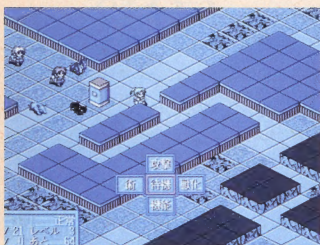




特集 SLASHの活用



ぶたさん



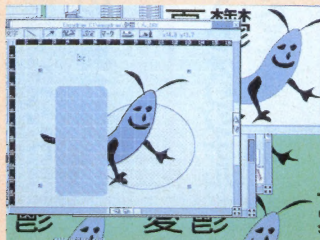
ダイアット・ヴァークス



PhotoCDとX68000



ハードコア3Dエクスタシー



Easydraw SK-68K

# Oh!X

## C O N T

### ●特集

## 33 SLASHの活用

- |    |  |      |
|----|--|------|
| 34 | SLASHと関連ツールから見た<br>3D処理の可能性を探る         | 中野修一 |
| 36 | 特殊機能のデバッグ<br>モデラの修正と拡張                 | 菊地 功 |
| 40 | モデリングの省力化のために<br>回転体生成プログラム            | 田村健人 |
| 44 | 面の順番を自動処理する<br>ポリゴンソートフィルタ関数SortPoly() | 丹 明彦 |
| 49 | 基礎からのSLASH<br>とりあえず三角錐を回してみる           | 山田純二 |

### ●カラー紹介

- |    |   |      |
|----|---|------|
| 15 | Oh!X Graphic Gallery<br>SLASHの活用        |      |
| 18 | 新製品紹介 SX-PhotoGallery<br>PhotoCDとX68000 | 荻窪 圭 |
| 20 | 3Dステレオグラム生成ツール<br>FLICKERとはなにか?         | 中野修一 |

### ●THE SOFT TOUCH

- |    |  |      |
|----|--|------|
| 22 | SOFTWARE INFORMATION<br>新作ソフトウェア/TOP10 |      |
| 26 | GAME REVIEW<br>ぶたさん                    | 柴田 淳 |
| 28 | ダイアット・ヴァークス                            | 須藤芳政 |
| 30 | AFTER REVIEW<br>ロボットコンストラクションR.C.      |      |
| 32 | TREND ANALYSIS                         |      |

### ●読みもの

- |     |                               |      |
|-----|-------------------------------|------|
| 138 | 猫とコンピュータ 第85回<br>ゴメンナサイの値段    | 高沢恭子 |
| 144 | X-OVER・NIGHT 第40話<br>SF時代に向けて | 高原秀己 |

### 〈スタッフ〉

●編集長／前田 徹 ●副編集長／植木章夫 ●編集／山田純二 豊浦史子 高橋恒行 ●協力／有田隆也 中森 章 林 一樹 吉田幸一 華門真人 吉田賢司 朝倉祐二 大和 哲 村田敏幸 丹 明彦 三沢和彦 長沢淳博 司馬 護 清瀬栄介 石上達也 柴田 淳 瀧 康史 横内威至 進藤慶到 ●カメラ／杉山和美 ●イラスト／山田晴久 江口響子 高橋哲史 川原由唯 ●アートディレクター／島村勝頼 ●レイアウト／元木昌子 ADGREEN ●校正／グループごじら





表紙絵：塚田 哲也

1993 NOV.  
11

# ENTS

## ●シリーズ全機種共通システム

113 THE SENTINEL

114 S-OSで学ぶZ80マシン語講座

伊藤雅彦

## ●連載/紹介/講座/プログラム

16 獅子 in CG わ〜ると[第30回]  
騎士

江口響子

58 SIDE A 視点を制し空間を把握せよ

丹 明彦

66 SIDE B ポリゴン描画のためのエッジ検出法

横内威至

72 大人のためのX68000 [最終回]  
新しい世界へ静かに発進

荻窪 圭

74 新製品紹介  
OS-9/X68000 UltraC&Professional Pack V1.1  
OS-9/X68030 Technical Tool Kit V2.4.5

中森 章

75 NEW PRODUCTS  
Easydraw SX-68K

丹 明彦

82 Compact搭載3.5インチFDDの2DD対応  
3.5インチFDDを改造する

中村隆生

87 ファイル共有の実験と実践(その3)  
電話回線を使った転送アプローチ

由井清人

94 (で)のショートプロローグ その50  
オモイコンダラ・プログラム

古村 聡

102 X68000用CARD DRV対応カードゲーム  
ネストール(Nestor)

高山忠信

104 Oh!X LIVE in '93  
渚のアデリーヌ(X68000・Z-MUSIC用)  
エロティカ・セブン(X68000用・Z-MUSIC用SC-55対応)

加藤 隆  
中田健一

108 Creative Computer Music入門(26)  
調性の誕生と和音の機能

瀧 康史

112 (善)のゲームミュージックでバビンチョ

西川善司

121 こちらシステムX探偵事務所 FILE-VI  
誤差の少ない三角形自由変形

柴田 淳

127 目指せジョイスティックの星(3)  
進化する目標を追い続けて

伊遊晃あきら

130 FISH.Xに続け!  
スクリーンセーバーのモジュールを作る

石上達也

135 AD PCMを使ったメディアコンバート  
CASSAVE.X/CASLOAD.X

原篠 誠

140 ANOTHER CG WORLD

江口響子

ペンギン情報コーナー……142

愛読者プレゼント……145

FILES Oh!X……146

質問箱……148

STUDIO X……150

編集室から/DRIVE ON/ごめんなさいのコーナー/SHIFT BREAK/microOdyssey……154

UNIXはAT & T BELL LABORATORIESのOS名です。

Machはカーネギーメロン大学のOS名です。

CP/M, P-CPM, CP/Mplus, CP/M-86 CP/M-68K, CP/M-8000, DR-DOSはデジタルリサーチ

OS/2はIBM

MS-DOS, MS-OS/2, XENIX, MACRO80, MS C, WindowsはMICROSOFT

MSX-DOSはアスキー

OS-9, OS-9/68000, OS-9000, MW CはMICROWARE

UCSD p-systemはカリフォルニア大学理事會

TURBO PASCAL, TURBO C, SIDEKICKはBORLAND INTERNATIONAL

LSI CはLSI JAPAN

HuBASICはハドソンソフト

の商標です。その他、プログラム名、CPU名は一般に各メーカーの登録商標です。本文中では"TM", "R"マークは明記していません。

本誌に掲載されたプログラムの著作権はプログラム作成者に保留されています。著作権上、PDSと明記されたもの以外、個人で使用するほかの無断複製は禁じられています。

## ■広告目次

アイビット電子	166
EAビクター	8
科学工芸研究所	158
カプコン	9
計測技研	168
サンワード	167(下)
J & P	表3
シャープ	表2・表4・14-7
九十九電機	164-165
ネオコンピュータシステム	167(上)
P & A	160-163
Beシステム	159
マイクロウェア・システムズ	10
満開製作所	11・157



# 先が面白くなる。

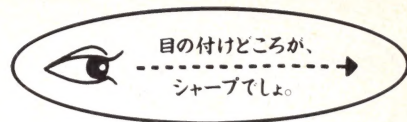
ウィンドウ環境のプラットフォームを確立、SX-WINDOW ver.3.0



●この画面は広告用に作成した、機能を説明するためのイメージ画面です。また、各種アイコン等は、SX-WINDOW ver.3.0がもつ機能を使って作成したもので、標準装備のものとは異なるものもあります。  
●本広告中のエディタで表示している文字のフォントはZeit社の、「書体倶楽部」のフォントを使用しています。



# SHARP



## に見たGUIの新展開。

- ① マルチフォントエディタ編集例。文字ごとに文字種、文字の大きさの指定、修飾が可能で、イメージデータの貼り付けもOK。
- ② CONFIG.SYSやAUTOEXEC.BATなどの編集に便利な「エディタ」モードの例。このように日本語マルチフォントエディタは、用途に合わせてカスタマイズできます。
- ③ ①の画面をプリンタで印字した例。対応プリンタも増えました。(カラー印刷は誤差分散により65,536色対応)
- ④ 「パターンエディタ」で作成したデータを、背景に設定できます。
- ⑤ バージョンアップした日本語フロントプロセッサASK68K ver.3.0の辞書メンテナンスがウィンドウ上で可能。
- ⑥ アイコンデータや背景データを作成する「パターンエディタ」。文字の貼り付けなど、編集機能も一段とフレンドリーに。
- ⑦ オリジナルに作成したアイコンパターンの例。
- ⑧ 512×512ドットの範囲内で65,536色の表示が可能。
- ⑨ さまざまなグラフィックフォーマットに対応しています。
- ⑩ 任意のサイズに縮小・拡大表示可能。
- ⑪ 異なる画像フォーマットへのコンバートができます。
- ⑫ 「CGAウィンドウ」、65,536色(最大)のコンピュータアニメーション表示が可能です。

発展性のあるプラットフォームとしてのウィンドウシステム、  
SX-WINDOW ver.3.0が提供する新たなGUI環境が  
さらなるウィンドウ時代を予見する——。

国産オリジナルウィンドウとしての意味、未来への確かなビジョン、  
ユーザーインターフェイスや高速化へのゆるぎない探求が  
ここに凝縮されています。

65,536色表示はもちろん、さまざまな画像フォーマット対応、  
イメージデータのコピー&ペースト、

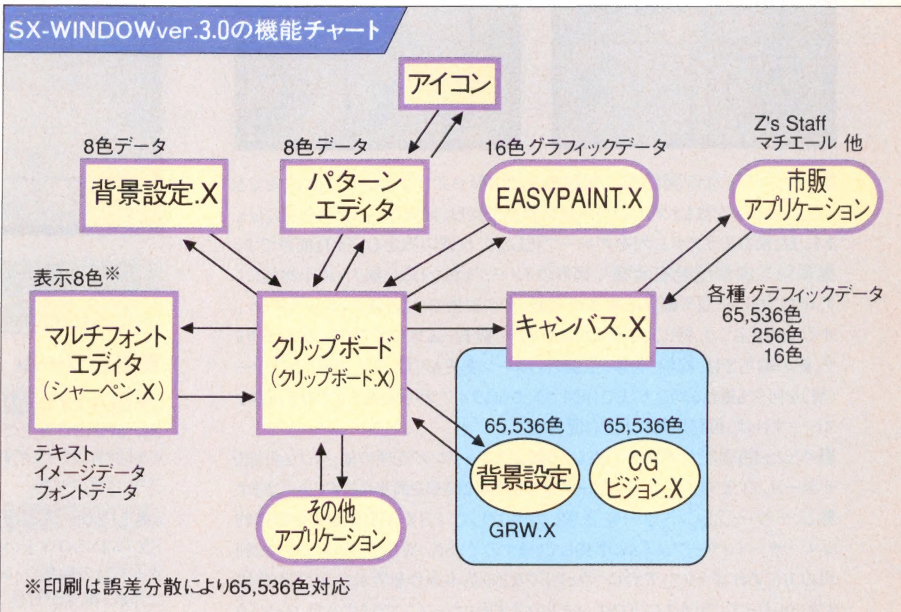
動画、音楽/音声再生をサポートするマルチメディア環境。

そして、何よりもこれらが密接に連携して

統合的にハンドリングできるエキサイティングな環境を創造しています。

未来を照準に入れたウィンドウアーキテクチャ、

そのインテリジェンスがいよいよX68030/X68000シリーズで享受できます。



**68030**  
32bit PERSONAL WORKSTATION

**68000**  
PERSONAL WORKSTATION・XVI

X68030

X68030 Compact

X68000 XVI

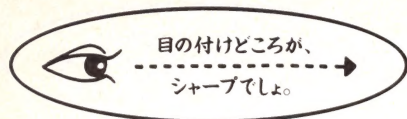
X68000 XVI Compact





# SHARP

## X68030/X68000シリーズ



# 成熟するウィンドウ環境で

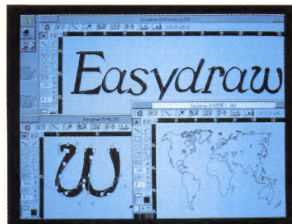
## SX-WINDOW対応ドローイングツール。

### Easydraw SX-68K

CZ-264GWD 標準価格19,800円(税別)

NEW

ホビーからビジネスまで幅広い分野で活用できる、待望のドローイングツールです。イラスト、フローチャート、地図、見取り図など各種グラフィックが製図感覚で作成できます。また作成したデータは他のSX-WINDOW対応アプリケーションでも利用でき、企画書やプレゼンテーション資料の作成をサポートします。



■スピーディな作図作業：製図感覚で図形や文字がスピーディに作成できます。一度描いた後もオブジェクト単位の移動や変形、回転なども素早く自在に行えます。また複数のオブジェクトをグループ化したり、位置の固定(ロック)も簡単です。

■多彩な編集機能を装備：図形のイメージを損なわない拡大・縮小機能により、レイアウトの確認や細部の編集が可能。文字編集では、各種フォント、スタイル、サイズが指定でき、特に文字サイズはポイント、級数、mm単位で任意に変更できます。線の編集では、線幅、矢印、点線のパターン変更も可能。また、透明なレイヤー(層)を何枚も重ねるような方法で作図でき、さらにライブラリを利用してそのデータをストックすれば、再利用時に大変に便利です。

■ベジェ曲線をサポート：点と点を結ぶスムージング処理の他、ベジェ曲線をサポートしていますので、少ないデータ量でも複雑な図形を簡単に描くことができます。

■ユーザーフレンドリーを追求したやさしさ：SX-WINDOWの標準的なユーザーインターフェイスに準拠していますので、SX-WINDOWをすでにご利用の方であればマウス、アイコン、ウィンドウなどの基本操作を学ぶことなくすぐに作図が始められます。作図ウィンドウは、メモリの許す限りオープンできますので、ウィンドウ間でのコピー&ペーストも可能です。

■豊富なデータ資産が活用可能：本ソフトで作成したデータを他のSX-WINDOWアプリケーションで利用できます。日本語マルチフォントエディタ「シャープペン.X」などにそのまま貼り込み、企画書などへの活用も可能。またサンプルデータを豊富に用意している他、「CANVAS PRO-68K」のドローデータ、「Easypaint SX-68K」のデータをそのまま本ソフトで利用することもできます。

■レーザープリンタドライバを付属：レーザープリンタ(ESC/Page, LIPS III, PostScript)の高解像度で美しい印刷が可能です。またこのドライバはSX-WINDOW対応の他のアプリケーションでも利用することができます。

※ESC/Pageはセイコーエプソン㈱の、LIPS IIIはキヤノン㈱の、PostScriptはアドビシステムズ社の登録商標です。

4MB, ver.3.0

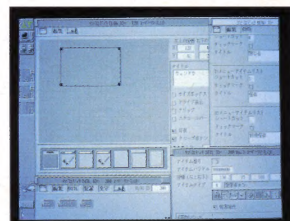
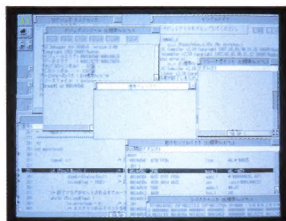
## 待望のSX-WINDOW開発支援ツール。

### SX-WINDOW 開発キット Workroom SX-68K

CZ-288LWD 11月発売予定

NEW

SX-WINDOW用のソフト開発に必要な開発ツールやサンプルプログラムを装備。プログラムの編集、リソースの作成、コンパイル、デバッグといった一連の作業をSX-WINDOW上で効率よく実行できます。初めてSX-WINDOW用のプログラムに挑戦する人にも、簡単に基本機能の理解ができる33種のサンプルプログラム付き。また各マネージャ解説と関数リファレンスの詳細なマニュアルも装備しています。※メインメモリ4MB以上、SX-WINDOW ver.2.0以上、C compiler PRO-68K ver.2.1が必要です。



### キット構成

#### 開発ツール

##### ●SXデバッガ

SX-WINDOW上で複数のプログラムを同時にデバッグすることができるソースコードデバッガ。

##### ●リソースエディタ

SX-WINDOW上のリソースをリソースタイプごとの編集ウィンドウでビジュアルに作成・編集が可能。

##### ●リソースシンカ

Cコンパイラやアセンブラで作成したリソースデータファイル(オブジェクトファイル)をリンクしてリソースファイルを作成。

##### ●サンプルメイク

サンプルプログラムのコンパイル作業をSX-WINDOW上から、XCver2.1のMAKE.Xを呼び出して、自動実行する簡易メイクユーティリティ。

#### サンプルプログラム

##### ●基礎編(23種)

各マネージャの基本的な機能のみを用いた基本動作の理解。

##### ●応用編(4種)

基礎編での基本機能を応用した簡単なアプリケーションの作成。

##### ●実用編(6種)

基礎/応用編での機能を駆使した、実用的なアプリケーションの作成。

#### ■その他ファイル

##### ●インクルードファイル

Cコンパイラとアセンブラ用の関数定義、データ定義ファイル。

##### ●ライブラリファイル

Cコンパイラ用関数ライブラリ。

#### マニュアル

- ユーザーズマニュアル
- プログラマーズマニュアル
- SXライブラリマニュアル



新製品

**68030**  
32bit PERSONAL WORKSTATION



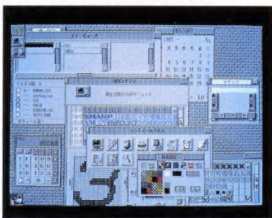
# さらに高度な創造次元へ。

- 65,536色対応、動画ウィンドウ標準装備。

## SX-WINDOW ver.3.0 システムキット

CZ-294SS(5インチ版)/CZ-294SSC(3.5インチ版)各標準価格19,800円(税別)

自然描写に迫る美しい表現が可能65,536色表示のグラフィックウィンドウを装備。さらにグラフィックウィンドウ内でのアニメーション動画表示、各種グラフィックデータのコンパートも実現しました。またイメージデータの貼り付けなどをサポートした日本語マルチフォントエディタを始め、クリエイティブワークを支援する数々の便利機能を装備、Human68k ver.3.0システムディスクも付属しています。  
※メインメモリ4MB以上が必要です。SX-WINDOW ver.1.0/1.1/2.0をお持ちの方には有償バージョンアップを行っています。



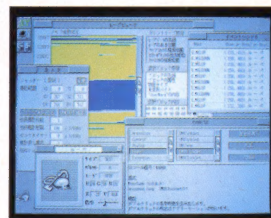
- 「SX-WINDOW開発キット」のサポートツール。

## 開発キット用ツール集

CZ-289TWD 12月発売予定

SX-WINDOW開発キットをさらに使いやすくなるためのツールです。SXコールの簡易リファレンスを簡単に検索するインサイドSX、イベントの発生を常時監視確認するイベントハンドラ、リアルタイムにメモリブロックの利用状況を表示するヒープビューアなど11種のツールが用意されています。

(2MB, ver.2.0)



- SX-WINDOWを楽しく使うためのアクセサリ集。

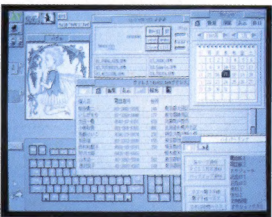
## SX-WINDOW デスクアクセサリ集

CZ-290TWD 標準価格14,800円(税別)

SX-WINDOWをさらに便利に、楽しく使うためのデスクアクセサリ集です。スクリーンセーバ、アドレス帳、電子手帳通信ツール、パズルなど12種類の豊富なアクセサリが収められています。

①キーノート②スクリーンセーバ③スクラップブック④ミュージックボックス⑤ハイパーリンク(電子手帳通信ツール)⑥アドレス⑦スケジューラ⑧ウィンドウアイコンフィア⑨ソフトウェアキーボード⑩パズル⑪ファイルサーチ(ファイル検索ツール)⑫フォントリカ。

(2MB, ver.3.0)



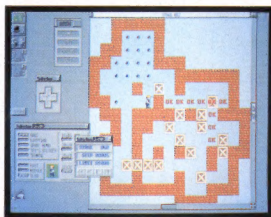
- SX-WINDOW対応になってさらにパワーアップ。

## 倉庫番リベンジ SX-68K ユーザー逆襲編

CZ-293AW(5インチ版)CZ-293AWC(3.5インチ版)各標準価格6,800円(税別)

10年にわたるユーザーの投稿など、新作306面が目白押し。まさに倉庫番の最強版がSX-WINDOW上で楽しめます。移動可能先が表示されるAI機能を搭載、またマウスをクリックするだけで簡単に問題を作成できるエディット機能や、キャラクタを替えてちよっと違った雰囲気ゲームが楽しめるキャラクタ変更機能も装備しています。半年で解いたらあなたは天才?です。

(2MB, ver.1.1)



- マルチタスク機能をはじめ、通信環境がさらに充実。

## Communication SX-68K

CZ-272CWD 標準価格19,800円(税別)

通信環境をさらに高めたウィンドウ対応の通信ソフトです。マルチタスク機能により他のアプリケーションソフトを実行中でも簡単に通信が可能。また、ホスト局をクリックするだけの自動ログイン機能、初心者にも簡単なプログラム機能、最新モデム(20種類)もフルサポートしています。

(2MB, ver.1.1)

- ウィンドウ対応グラフィックツール。

## Easypaint SX-68K

CZ-263GWD 標準価格12,800円(税別)

マウスによる簡単操作、65,536色中16色の多彩な表現、クリエイティブマインドに広がるウィンドウ対応ペイントツールです。同時に複数のウィンドウを開いて編集でき、各ウィンドウ間でのデータ交換もできます。

(2MB, ver.1.1)

- FM音源サウンドエディタ。

## SOUND SX-68K

CZ-275MWD 標準価格15,800円(税別)

他のミュージックソフトで演奏中の音色を、簡単に作成、変更できるマルチタスク機能、またエディット、イメージ、ウェーブの3つの編集/確認モードを装備。作成中の音色も50曲の自動演奏でリアルタイムに確認、編集できます。まさにミキサー感覚で音創りが楽しめるツールです。

(2MB, ver.1.1)

**PRO-68K**

シリーズ

- X68030/X68000対応

NEW

**COMPILER PRO-68K**  
ver.2.1  
NEW KIT

CZ-295LSD 標準価格44,800円(税別)

※メインメモリ2MB以上が必要です。

※C compiler PRO-68K/ver.2.0/ver.2.1をお持ちの方には有償グレードアップサービスを行います。

C compiler PRO-68KのX68030/X68000対応版、MPU68030、MC68882の命令セットに対応したアセンブラ、デバッガ、ソースコードデバッガを付属。またHuman68k ver.3.0、ASK 68K ver.3.0にも対応。新たにGPIOライブラリ、MC68882対応フロッピーライブラリを付属しています。



※ (2MB, ver.1.1) の表示は、メインメモリ2MB以上、SX-WINDOW ver.1.1以上が必要であることを示します。

※発売予定のソフトの画面は実物とは異なる場合があります。



オリジナル・テレホンカードプレゼント中

詳しくはパッケージの中に入っているユーザー登録はがきをご覧ください。

いっくぽ〜ん♡

闇に閉ざされた妖精界を舞台に、  
お調子者の魔法使いと、しっかり者  
の妖精の凸凹コンビが繰り広げる、  
痛快、パカポコ ファンタジー シュ  
ーティング

COTTON

コットン



BIGでグレートな

- ΔV68000シリーズ  
(ΔV68030 対応)
- ハードディスク対応

ある日のこと、突然、闇の波動が光のプリズムを狂わせ、世界中の光をすいこみ始めた。それから、闇の夜が続き、眠ったままの朝を目覚めさせる方法がわからないまま、人々は不安な生活を送っていた。いっぼう、そんなことにはまるっきり無関心な、食いしん坊魔法使いコットンは、妖精シルクからのお願いも知らんぷり。ところが、魔物を退治すると大好物のWILLOW(ういろう)が手にはいると聞いて、お目々キラキラ!とにもかかわらず、コットンのWILLOW探索の旅(?)が始まります……が?

'93年9月24日発売

¥9,800(税別)

5"FD×2枚組

3.5インチ・ディスク・ドライブを  
ご使用の方へ

5インチ版をご購入の上、そのフロッピーディ  
スクを当社、カスタマーサポート係宛に郵送  
してください。3.5インチ版ディスクを返送いた  
します。(送料当社負担)

©1991,1993 SUCCESS

通信販売：当社の製品をお近くのパソコンショップでお買い求めになれない場合、通信販売もご利用いただけます。  
商品名、機種名、住所、氏名、電話番号を明記の上、右記住所まで定価プラス3%消費税分を現金書留にて通信販売部  
宛にお送りください。(送料当社負担)

エレクトロニック・アーツ・ビクター株式会社

〒150 東京都渋谷区神宮前2-4-12 フルークス外苑

製品に関するお問い合わせ：03-5410-3100(月～金、13:00～16:00)



CAPCOM

△68000

格闘の頂点をめざす者たちへ。

# STREET FIGHTER II

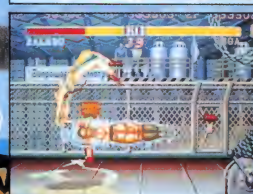
カプコンが、パソコンを熱くする!

**X68000用ソフト第3弾**  
**ストリートファイターIIダッシュ**  
**11月26日(金)発売予定 予価12,800円**

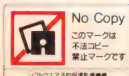
●X68030対応 ●要2メガバイトメモリ以上 ●MIDI対応

**CPSファイターパソコンアダプター付**

あのCPSファイターでプレイできる!



©CAPCOM 1991, 1993 ALL RIGHTS RESERVED.



株式会社 **カプコン** 国内営業本部/〒540 大阪市中央区釣鐘町2-2-8 東京支店/〒163-02 東京都新宿区西新宿2-6-1 新宿住友ビル43F  
★カプコンソフト情報★ 大阪(06)946-6659 東京(03)3340-0718 札幌(011)281-8834 仙台(022)214-6040 名古屋(052)571-0493  
広島(082)243-6264 松山(0899)34-8786 福岡(092)441-1991 — 電話番号は、よく確かめておかけ間違いのない様にしてください。 —



# OS-9/X68030 V2.4.5

microware®

32bit.  
PERSONAL WORKSTATION

0

1

TIMER

HD BUSY

## OS-9のX68030対応版、新登場。

68系のリアルタイム・マルチタスク・オペレーティング・システムOS-9に、シャープX68030対応最新版が登場。UNIXライクな操作性と洗練された機能で、X68030の機能を最大限に引き出します。

- リアルタイム・マルチタスクOS
- マルチウィンドウをサポート
- 最大10ユーザーのマルチユーザ環境
- 大容量SCSIハードディスクをサポート
- テキストエディタμMACSを標準装備
- VJE-γ V2.0による快適な日本語入力



価格  
¥25,000(税別)



### ●OS-9/X68030シリーズラインアップ

価格(税別)

OS-9/X68030 V2.4.5	¥25,000
Ultra C & Professional Pack V1.1	¥45,000
Technical Tool Kit V2.4.5	¥20,000
X Windows V11.5	¥30,000 8月末
MPFMフル動画 V1.0	未定 開発中

※ソフトウェアの内容・仕様は、改良のため予告なく変更する場合があります。

※OS-9は、マイクロウェア・システムズ(株)の登録商標です。

※X68030は、シャープ(株)の登録商標です。

※VJE-γは、(株)バックスの登録商標です。

※その他製品名、会社名は、各社の登録商標または商標です。

※この製品の無断複製、レンタル等は、法律によって禁じられています。



夢は、いまだきよ……

コンパクト XVI 改造機。  
弊社にて1年保証。クロックは10/16/24の3モード。16/24MHzは背面トグルスイッチにより切替。RED ZONEの24MHzでは正常動作しないソフト等がありますが、10/16MHzでご使用になれます。

△68000 Compact XVI 改

**RED ZONE** ¥160,000



・シャープ製CZ-6FD5  
完全コンパチブル・オートイ  
ジェクト機能付・ドライブ番  
号切替スイッチ付・木製（ナ  
ラ材）フロントパネル・対応  
機種/CZ-674C/30  
0C/310C/500C/  
510C  
・カラーリングオプションは  
プラス5,000円です。

①②③④⑤⑥

シャープ製CZ-6FD5  
完全コンパチFDD(MK-FD1)

満開式軟盤駆動装置壱號  
¥39,800(税別、カラーモデル¥44,800)



新発売

**TOWERJACK**

満開式拡張型  
硬盤駆動装置壱號

1ギガバイトHD+SCSIスロット×3のミニボックス  
(MK-HD1-EX)特別価格 ¥180,000(税別)

○HDドライブは東芝製。さらに接続用デバイスも続々  
発売予定。

**RED ZONE**

+

**MK-FD1**

特別価格

→ **¥180,000**(税別)

(カラーリングモデルは+¥5,000)

ごめんなさいの価格改定98バスマウスアダプタ

**MOUSEJACK68-98**

(MK-MJ1)

¥4,000(税別)

当ショップは通販専門店です。X680×0用各種ハード・ソフト  
も取り扱っております。お電話にて商品リストと注文書をご請  
求ください。RED ZONEのご購入には承諾書が必要です。  
合わせてご請求ください。

〒171 東京都豊島区長崎1-28-23 Muse西池袋2F  
TEL (03)3554-7441 FAX (03)3554-3856

**パソコンショップ満開**  
**(株)満開製作所**



# Mac



**SOFT  
BANK**

ソフトバンク株式会社 出版事業部  
〒103 東京都中央区日本橋浜町3-42-3 TEL.03-5642-8100



Macintoshユーザーの  
創造力向上マガジン

# User

11月18日創刊

特集  
1

PowerPC時代を生き抜く最強のMac  
我がMac選びに悔いなし

特集  
2

INIT,cdevで操作性を高める  
3万円でできるKT7チューンアップ

創刊特別2大付録

1 CD-ROM……MacBin

市販ソフト体験版およびオンラインソフト満載! その他もりだくさん

2 別冊…CD-ROMドライブ購入ガイド

月刊マックユーザー／毎月18日発売／定価980円(税込)

◆MacUserはZiff-Davis Publishing社との提携誌  
Ziff-Davis Publishing社は世界最大のコンピュータ専門出版社  
43万人の読者を有する米国版MacUserと9万人の購読者を持つMacWEEKとの提携により  
正確かつ新鮮な情報を確実に報道していきます

※創刊号は完売する場合がありますので、書店でお早めにお求め下さい



Oh/X Books改訂版

# Z-MUSIC システム Ver.2.0

ついにMUSICシステムの正式バージョンアップ版が登場します。  
X68000の音源ドライバとしてさらに使いやすく高機能なものになりました。

## ver.1.0/1.1からのバージョンアップ内容

PCM8対応AD PCM同時発音8声音量可変  
モジュレーション用波形メモリ搭載  
PCMバンクに対応  
ステップエディット系コマンド追加  
X68030完全対応ユニバーサルバージョン  
RS-232C対応版収録

POLYPHON対応版収録  
再生専用機能縮小版収録  
Cコンパイラ用ライブラリ完成  
AD PCM加工機能強化  
さらにクオリティを高めたAD PCMデータ  
もちろん、全ソースプログラム付属&ライセンスフリー



5"2HD 6枚組  
予価5,000円(税込)

**SOFT  
BANK**

ソフトバンク株式会社／出版事業部

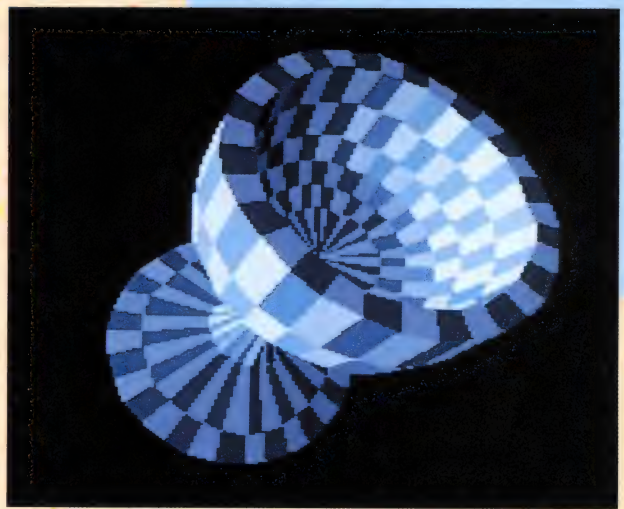
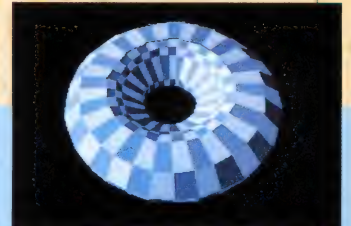
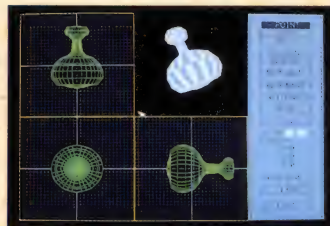
〒103 東京都中央区日本橋浜町3-42-3 TEL03-5642-8100



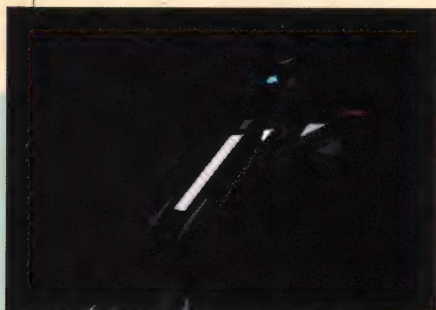


▲連載「ハードコア3Dエクスタシー」から、路面表示のサンプルプログラム。背景にビルも加え、ある程度「それっぽい」作りになっている。まだマップシステムが組み込まれていないため、コース全体をまとめて回転させている。10MHz機だとかなり重い。

今月はCGA講座がお休みなので、SLASHを使ったサンプルプログラムの数々を紹介します。SLASHも周辺ツールも元気に成長しつつあります。

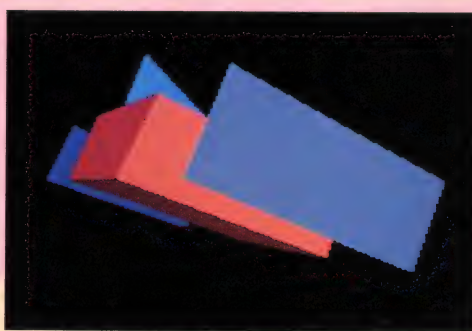
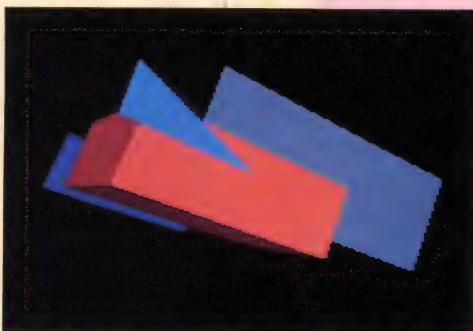
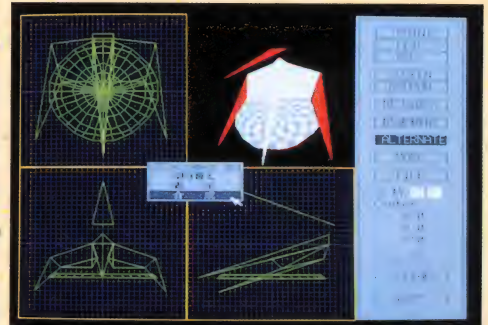
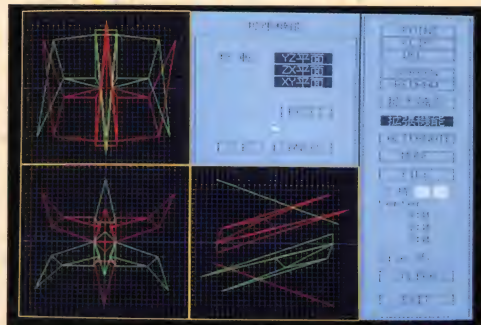


▶回転体生成プログラムで作成されたワイングラス。このように、壺やドーナツなどの回転体は簡単に作成できる。ある程度までなら面の順序も自動的に調整する。ちょっと細かすぎるのでリアルタイムゲームを主眼としたSLASHにとっては重いデータだが、かなり複雑なデータでも破綻なく表示してくれた。



▲オブジェクト爆発プログラムのひとコマ。ポリゴンデータを書き換えている。

▶デバッグ&拡張されたSLASHのモデラ。裏画面とのオブジェクト合成や拡大、対称といった便利な特殊機能が問題なく使用できるようになった。これで大きなバグはほとんどなくなっている。



◀自動的にポリゴン定義順序を並べ替えるSortPoly()関数の実行例。面の定義順がおかしいと左のような表示になってしまうことがあるが、この関数を使うことで可能な限り補正することができる。



# 響子inCGわ〜るど

「悪い、悪い、待たせたなー。許せ！」

「おまえねー、もう少し女らしい言葉使ったら」

「しょうがないじゃん。地なんだから」

これが僕の彼女。髪は刈り上げのベリーショート。身長は僕よりちょっと低めの168センチ。やせてて、足が細いのはいいんだけど、胸がべちゃんこなんだ。

顔はアイドル系なので、学校のセーラー服を着ると結構かわいい……でも、今日の格好じゃあ、後ろから見たら男が2人で歩いているみたいだ。いや、前からだってそう見える。

すりきれたGパン。兄貴のおさがりだっていう

黒のTシャツ。玄関にちょうど脱いであった、という理由だけではいてきた弟のスニーカー。

「おまえねー、もう少し女らしい格好したら」

「しょうがないじゃん、服に興味ないんだからさ」

日曜日の秋葉原。雨が降っていて肌寒い。不景気のせいか、人どおりはそんなに多くない。が、僕らの目指すところだけはごった返していた。

中古ゲームソフト店。品数は、まあ普通ってところかな。ここの売り物は、毎月第3日曜日に行われるオリジナルゲーム。誰でも参加できる。高校野球みたいなトーナメント戦になっていて、優勝者は、店内にある好きなゲームソフトを両手で持てるだけもらえる。負けたら、最低1本ゲームを買わなくてはいけない。

オーナー兼店長は、某社のゲームデザイナーだった。いまは、自分の作りたいゲームを基板もろとも趣味で作っている。彼の作るオリジナルゲームは、間違いなく面白い。しかも、ここでしかプレイできないし、勝てば商品だってもらえちゃう。第3日曜日は、いつも腕試ししたいやつらであふれている。

今度のゲームはドッジボールにすごく似てた。どこから飛んでくるかわからない岩を、剣で粉々に打ち砕くか、盾で跳ね返す。さもなくばよける。体に当たるとダメージを受け、それがマイナス点となって持ち点から差し引かれる。持ち点が0になるまでの時間を競うんだ。

小学生のとき、僕はドッジボールが苦手だった。なんというか、空間の感覚がうまくつかめないんだ。それはいまも変わらない。だから、ゲームのテストプレイの結果はさんざんだった。友達に商品のゲームソフトをガッポリ持って帰ってくるぜ、なんて約束しなければよかった。

「こんなんじゃ、1回戦で負けちゃうよ。どうしよう」







彼女に向かってつぶやいた。

「しょーがねーなー」

彼女は僕を引っ張って、店の隅に連れていった。

「Gジャンと帽子を貸してみ」

こうして、僕と彼女は入れ替わった。

ゲームが始まった。1回戦の3組目で僕の名が呼ばれ、彼女は低い声で「はい」と返事をした。店の中は薄暗い。だから、プレイヤーがテストプレイのときと違う人間で、しかも女の子と気づくやつはいなかった。

結果は……僕の、いや彼女の優勝だった。ドッ

ジボールにメチャクチャ強い女の子って、学年に必ずひとりはいたよね。彼女はそのタイプだった。

外に出ると、いつの間にか雨は上がり、青空が見えた。彼女が見えない刻で、垂れ込めた雨雲を切り払ったかのようなだった。雲の切れ目から、黄色い秋の日差しがこぼれ、彼女はさっさと歩いていった。

僕はゲームのいっぱい入った袋をわきにかかえて、彼女のあとを追いかけた。そして、思ったのだ。騎士に助けられたお姫さまって、こんな気分なのかなあと。



# PhotoCDとX68000

Ogikubo Kei

荻窪 圭

写真をCD-ROMで保存すると、画像が劣化しない、パソコン上で加工できる、などのさまざまなメリットがあります。このほど、その「PhotoCD」のX68000用ビューアが計測技研より発売されました。



X68000にCD-ROMドライブをつなぐ。それで何ができるのか。CD-ROMというのは一般人レベルでは自分で作成することができないわけで、どっかのサードパーティなりシャープなりがCD-ROMに収めたソフトを開発してくれない限り、利用範囲は非常に限られる。たとえば、Macintosh用やPC用のCD-ROMデータ集を買ってきてX68000にコンバートしてみたり、辞書モノのCD-ROMにX68000からアクセスしてみたり、どうもいまひとつ購入意欲がわかないものがあったりする。

が、自分でCD-ROMを焼けるとなると話は変わる。何の話かっていうと、PhotoCDである。PhotoCD。

## PhotoCDの基礎の基礎なのだ

PhotoCDはコダック社が開発したフォーマットである。35mmフィルムで撮った写真のフィルム（要するに、現像済みの35mmフィルムなのだが）をフィルムスキャナ（だと思う）でスキャニングしてデジタル化し、ライトワンスのCDに焼いてくれる、って代物だ。日本では昨年の10月から始められたサービスだ。そもそもの目的は写真をデジタル化して保存し、テレビをビューアとして鑑賞しようというもの。ちゃんと紙焼き用の高解像度データももっているから、印画紙に焼くこともできる。なか

なかおいしい代物なのである。

ノーマルなPhotoCDの場合、100枚までデータを入れることができる。どうして100枚しか入らないか。絵1枚につき、5種類の解像度で収められているからだ。インデックス用、モニタ表示用、印画紙用、ってな感じ。解像度は下から順に、192×128、384×256、768×512、1536×1024、3072×2048となっている（写真1）。この5種類のなかから必要な解像度のファイルを読み込めばいいわけだ。

## PhotoCDを作るのだ

てなわけで、なかなかX68000で使う話にならないが、それはもうちょい待て、ってことで、PhotoCDを作る話へいく。

PhotoCDを作るのは簡単だ。撮影済みの未現像フィルムかネガフィルムを持ってコダック社のプリントを扱っている写真屋さんへ行き、「PhotoCDにしてちょ」といえばいい。その写真屋さんがオオボケでなければ、それで理解してくれる。100枚まで入るから、24枚撮り4本とかやるとお得。

が、PhotoCDの場合、あとからデータを追加することができる。とはいえ、普通に考えてみればいいのだが、CDでは一度書いた部分はもう書き直せない。つまり、FATに当たる部分は1回書いてしまえば終わるわけ、追加書き込みということは、パーティションを切る、みたいなイメージになる。これをマルチセッションというのだが、パソコンでPhotoCDをマウントする場合、ドライブがマルチセッションに対応していなければならないわけで、そこは念頭に置いておこう。

では、36枚撮りフィルム1本をPhotoCDにすることを考える。その価格だ。

まず基本料金が500円。ディスクが1枚1,000円ということになっている。街でライトワンスのブランクCDを買って3,000円くらいだ（らしい）から、まあ、リーズナブルだ。で、写真を1枚書き込むたびに、未

現像フィルムからなら80円。現像済みネガフィルムからなら100円。

36枚撮った未現像フィルムを直接PhotoCDにする場合は、 $500 + 1,000 + 80 \times 36 = 4,380$ 円となる。これが高いか安いかは人それぞれだろうが、私はけっこうリーズナブルではないかと思っている。スキャナを買うと場所をとるし、10万円以上とんでっちゃうし、印画紙からスキャンするわけで、どうしても画質的には荒れる。きれいに撮るにはフィルムスキャナが欲しいところなのだが、こちらは20万円以上かかるのだ。PhotoCDの場合、画質は悪くないし、場所もとらないし、10万円分作ろうと思ったら、けっこうな枚数になる。もっと安いにこしたことはないのだけれどもさ。

肝心の納期だが、コダックのラボでの作業が5日かかる、ということになっている。土・日は休みであるから、5日というより、5営業日というほうが正しい。実質的には、1週間みておくのが安全だ。

## X68000でPhotoCDを見る

X68000は16ビットカラーの出るパソコンであるから、十分リアルにPhotoCDを楽しめる。

まず、CD-ROMドライブを用意する。これはそのへんのSCSIのCD-ROMドライブであればつながるだろう。今回はエレコム製のFixell CD-ROMドライブをつないでみた。東芝の倍速ドライブを搭載してあってけっこう速い。倍速ドライブ、ってのは、特に説明いらないよな。まあ、計測技研の出しているドライブを使うのが賢明だろう。

続いて、CD-ROMのドライブとPhotoCDのビューアだが、計測技研からSX-PhotoGalleryってものが出た。基本セットで15,800円もする代物だ。これで15,800円ってのが凄いが、まあ、いいや。

まず、こいつから、CD-ROMドライブをインストールする。と、ドライブがハードディスクの指定したディレクトリにコピー

写真1 解像度の低いほうから3つ並べてみた。これは新幹線（っていわれんでもわかるか？）。中央上のウィンドウがSX-PhotoGalleryだ





され、それがCONFIG.SYSに追加される。その際、CD-ROMドライブのSCSI IDを入力しなければならぬ。

お次は、PhotoGalleryをインストールする、っていっても、これはディレクトリごとコピーしてやればいい。

ではでは、セッティングが完了したと仮定して、PhotoCDをドライブに挿入する。と、CD-ROMのアイコンが現れる。

PhotoCDを見てみよう。

まずPhotoGalleryを起動する。グラフィックウィンドウが開いていないと、テキスト画面にPhotoGalleryが開いて8色に変換された写真を涙して見ることになるから、グラフィックウィンドウをまず開いておく。

お次の操作が傑作だ。最初、どうやりやいいんだかわかんなくて、思わずマニュアル(といっても、シャープペン.Xで読むオンラインマニュアルだけなんだけどさ)を読んってしまったくらいだ。

なんと、PhotoCDのアイコン(つまりドライブのアイコン)をPhotoGalleryのウィンドウヘドラッグするのである(写真2)。をを。

PhotoGalleryのウィンドウは写真を見るためではなく、写真のインデックスを見るためにある。要するに、目次だ。だから、ウィンドウにn枚中m枚目と書いてある。

では、次の写真を見る、あるいは前の写真に戻るときはどうしたらいいか。

これはもう凄いでっせ。なんと、横スクロールバーを使うのだ。横に長いフィルムをスクロールして見る、ってメタファーなつもりだろうけどさ、ちょっと違うんでないの、お兄さん。

気を取り直して、インデックスから気に入った写真をちゃんと見ようと思う。それには、右ボタンポップアップメニュー(写真3)でもって、ビューアーを起動する。と、その写真のウィンドウが表示されるが、まだインデックス用の192×128ドットのままだ。だから、ポップアップメニューで、解像度を上げる。384×256か768×512か。その上の解像度には対応していない。解像度を上げると、その解像度の写真が読み込まれるわけで、非常にきれいだ。写真が縦位置で撮られたもの場合は、回転機能を使う。(写真4)

さて、これをハードディスクにセーブしておいて、「MATIER」などのグラフィックソフトで加工して遊びたい。だが、このビューアーには保存の機能がない! ほんとにないのだ!

まず絵をコピーし、しかるのち、キャンバス.Xを起動して、ペーストするのだ。凄くい慢なこと。なお、トリミングなんて贅

沢なものもない(最低!)

で、キャンバス.Xで一度X68000のグラフィックデータにすれば、あとは煮て食おうが焼いて食おうが自由だ(写真5)。

ビューアーのユニークなユーザーインタフェースについて、追加しておこう。なんとなんと、ビューアーのウィンドウの拡大縮小は、SHIFT+ドラッグで行うのだ。あれ? 拡大縮小はちゃんとズームボックス使うんではなかったっけ?

\* \* \*

てなわけで、PhotoCDは非常に面白い。普通の35mmフィルムで撮った映像がそのままCD-ROMになって、そいつをハードディスクにコンバートしてやれば自在にさわれるのだ。

しかし、この、SX-PhotoGalleryには愛がない。低機能+謎のユーザーインタフェースだ。せめて、MATIERとZ'sSTAFF用のフィルタがあっても罰は当たらないはずだ。SX-WINDOW上にまともなグラフィックツールがない現在、Human 68k上のソフトもサポートするのは当然であり、それをしないのなら、PhotoCD専用ではなく、SX-WINDOW用CD-ROMツールキットみたいにして、PhotoCD、音楽CD、ISO9660、HFSなどなどを網羅したツール集にすべきだろう。ちょっと残念であった。

写真2



写真3



写真2 ドライブアイコンをこのようにPhotoGalleryへドラッグする  
写真3 ビューアーのポップアップメニューからは解像度や回転、コピーなどを選べる

写真4



写真5



写真4 縦位置の絵は回転させて見るとよい  
写真5 キャンバスへペーストしてみたビールのある風景と、縦に伸張して遊んでみた、ボールを追う子供の風景。それにしても、512×512ドットって、狭い(泣)



# 3Dステレオグラム生成ツール FLICKERとはなにか?

Nakano Shuichi 中野 修一



ちょっと変わったプログラムを紹介しましょう。新しくなったMATIERに付属するFLICKERはカラーの3Dステレオグラムを作成するためのツールです。マウスひとつで操作は簡単。画面の中に図形が浮かび上がります。オリジナルのステレオグラムも手軽に作成できます。視差間隔が自由に調整できますので、立体視のできない人には練習用にいかもかもしれませんね。

FLICKER.Xというのはカラーの3Dステレオグラムを作成するためのツールで、MATIER ver.2.0のおまけとしてついてくるものです。

本来なら今月あたりで新しいMATIERの機能を紹介すべきところですが、ざっと見たところ描画機能自体に関しては大幅な変更点はないようです。目玉はスキャナやプリンタなどのSCSI対応やタブレットの絶対座標対応ほか、周辺機器関連のサポートが充実したということでしょう。細かい部分ではいろいろ改良されている箇所があるようで、使い勝手がかなり向上しているようです。ということで、グラフィックツールの使いこなしとなると、やはり川原由唯氏にお任せしましょう。じっくり使い込



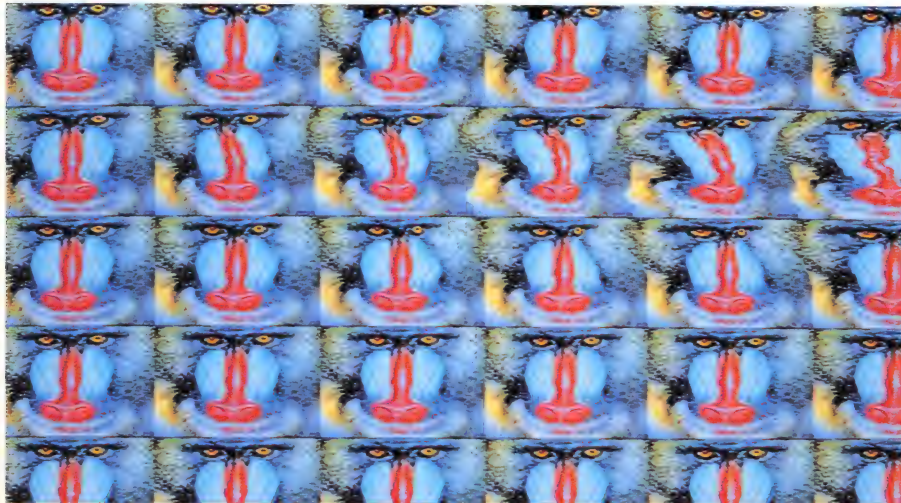
このように指定する

んでもらっていますので来月号にご期待ください。

で、このFLICKERというのはオマケツールなわけですが、どうしてどうしてかなり遊べるツールです。

以前のおまけについていた動くZOというのもそれなりに使えるツールではあったのですが、アニメーションの作成となるとDōGA CGAシステムの整備された環境ばかり目についてしまい、地味な存在となっていました。その点、今回のFLICKERのユニークさは際立っています。

地紋となるパターンはPICファイル、立体となる画像はIMGファイルで用意しなければなりません（もちろん標準でも用意されていますが）。あとはそれを組み合わせて実行するだけという簡単操作です。



実行結果。3Dペイントの文字を立体化した。難易度高！

背景となる図形を微妙に変形していくわけですが、若干右端にしわ寄せがいくようで、アルゴリズム自体には改良の余地ありといったところでしょうか。

## 平行法と交差法

FLICKERの立体視では、もちろん平行法と交差法が選択できます。

一般に簡単なのが平行法、難しいのが交差法といわれています（人にもよりますが）。遠くを見るように両眼の角度をあわせ、眼球の焦点だけは手前にあわせるようにすると平行法となります。逆に寄り目にして、焦点だけは少し遠目にあわせると交差法になります。

今回のFLICKER程度の画像ならどちらでも大差ないのですが（前後関係が入れ替わるだけ）、ステレオ写真などでは交差法でない大きな図形は立体視できません。

平行法では立体視できるものの大きさに限界があります。両目の間隔に依存する部分が大きいからです。人間の目というのは内側に向けるのは簡単でも外側には向きにくいものです。それでも徐々に慣らしていけば、21インチディスプレイで横幅120（最大）に広げても平行法で立体視できるようになりますから、人間の目もちょっとくらいは外側を向くようにできているでしょう。

## 背景画の描き方

基本的には、背景になるものはなんでもかまいません。絵はPICファイル（大きさは任意）ですので手持ちの画像データならなんでも使えます。ただし、あまりにのっぺりした画像だと変化がわかりませんのである程度は複雑なもののほうがよいでしょう。

普通に描かれた絵だと画像の歪みが目立ってしまい、できあがった画像が不自然になったり、立体視しなくても見当がついて



しまったりすることがあります。そうならないようになるべく無機的な絵を選んだほうがよいようです。その極端な例がランダムドットです。

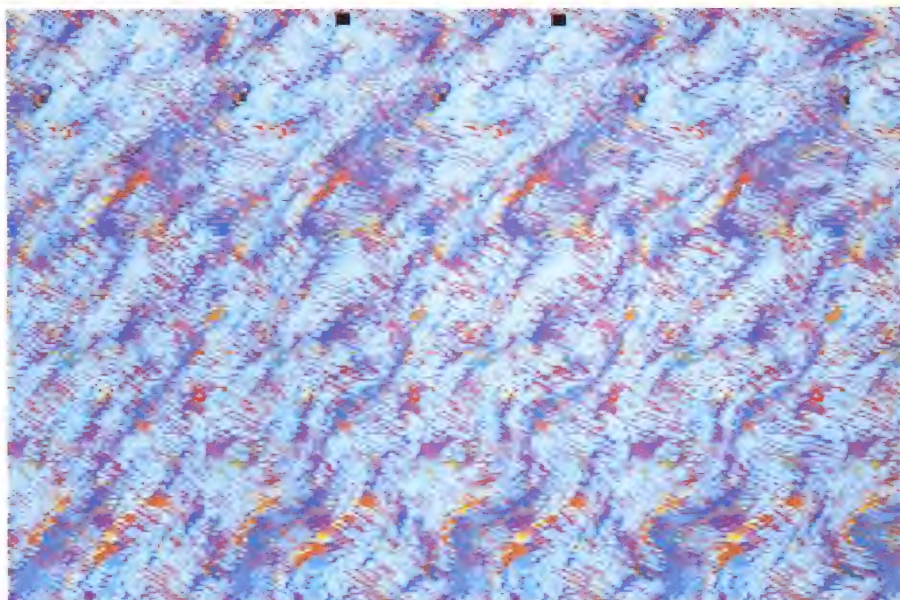
この場合、上下左右がつながるような図形を用意しておけばより自然な感じに仕上がるでしょう。MATIERなら矩形範囲のスクロールができますので、これとブラシを組み合わせれば継ぎ目のない模様を作るのもさして困難ではありません。

## 立体画像の作り方

立体画像をちゃんと作るのは面倒です。本来なら3Dモデリングされた物体からデータを生成したり3Dデジタイザがほしいところですが、とりあえず目分量で手描きすることになります。まあ、グラデーションでちゃんと面取りしてやればいだけなのですが……。

Z'sSTAFF PRO-68Kの登場以来、グラデーションはX68000の得意技ですので、MATIERでも関連する機能は充実しています。

メニューのいちばん上にあるグラデーションメニューを右クリックすることでグラデーションのパターンを選択できます。必要に応じて円グラデーションなどを選択するとよいでしょう。簡単に球体が描けます。ただし、円グラデーションでは速度（グラデーションの変化割合）をつけることがで



難易度低。平行法、交差法とも可

きないので円錐などを描くのは難しいかもしれません。

データはグレイスケールで作成します。後ろが黒で手前が白です。

コツとしては、最初に画面全体を赤なり青なりの色で塗っておきます。線画で軽く下描きをしておもむろにグラデーションで埋めていきます。最後に背景を黒にしてできあがりです。最初はあまり凝ったものは作らないほうがよいでしょう（経験者は語る）。

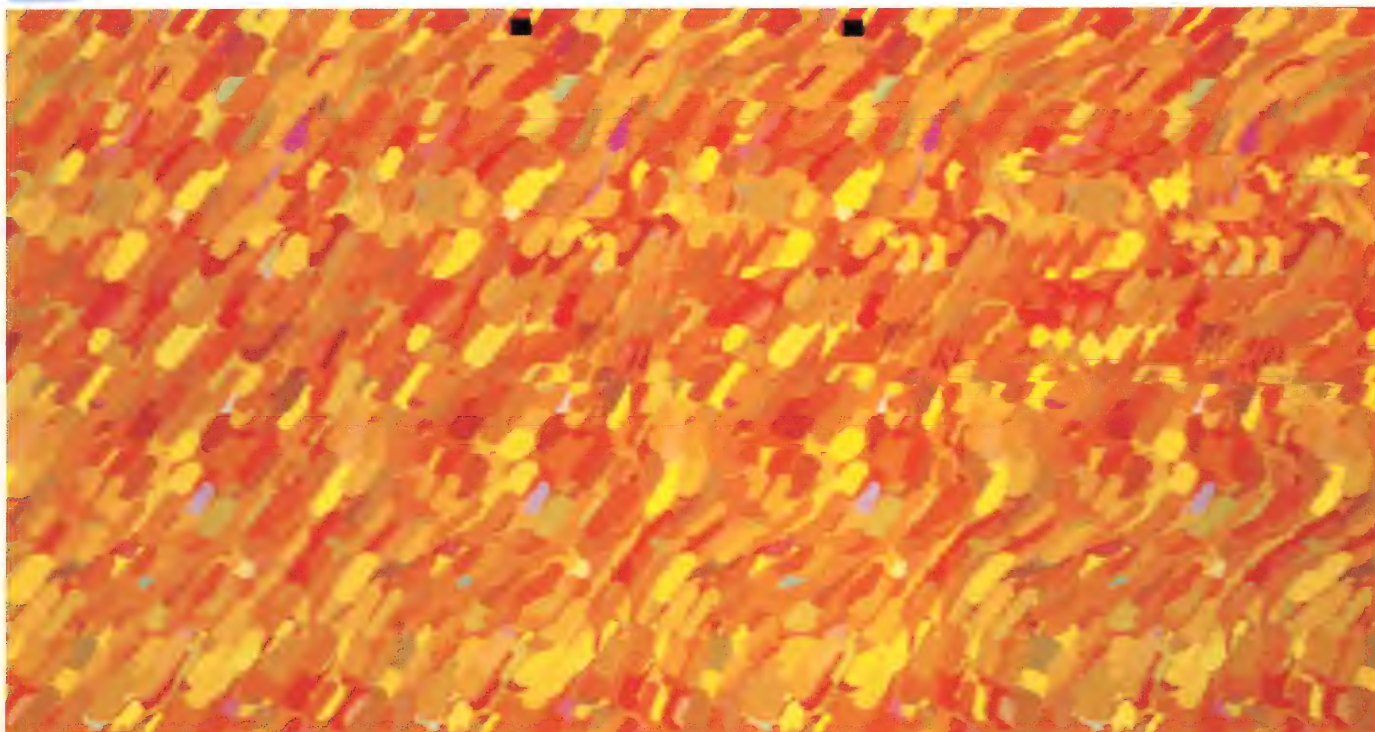
\* \* \*

最初は「左右の視力がかなり違うから」とか「こんなんじゃ焦点があうわけない」とか思っていた立体視も、一度成功するとヤミつきになってしまうものです。

そういえば、DōGAのビデオにもランダムドットのアニメ(?)が延々と収録されていましたが、カラー画像でやればもっともっと楽しそうです。すっかりメジャーになった立体視も単に眺めるだけでなく、もっと応用することを考えるべきかもしれませんね。

サンワード

(MATIERに付属)



難易度高。平行法のみ可



## SOFTWARE INFORMATION

前作「ストライダー飛竜」から、ちょうど1年。待ちに待ったアレの発売日が決定しました。封印中の人にはちょっと酷な時期かもしれませんが。発売まであと1カ月です。カレンダーに印をつけて待っててね。



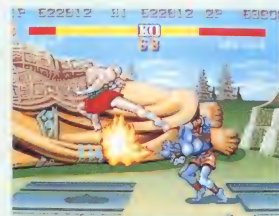
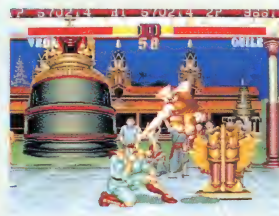
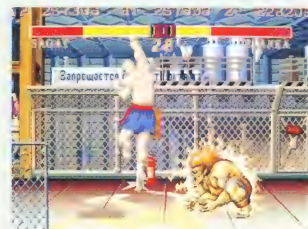
### ストリートファイターⅡダッシュ

ストリートファイターⅡ(以下、ストⅡ)。いわずと知れた対戦格闘ゲームの代名詞、ちょうど1年前にカプコン参入が決まったときから誰もが当然のように期待していた作品である。

今回移植されるストⅡはストⅡのバランスを対戦重視に調整し、プレイヤーキャラクターを増やしたものだ。以降、ストⅡ'turbo、最新作のスーパーストⅡとシリーズは続くが、トータルバランスで最も定評があるのがこのストⅡではないだろうか。

ストⅡは格闘ゲームの黄金時代を築いた歴史的な作品である。以後、各社のあと追により山ほどの格闘ゲームが発表されてきたが、ことキャラクターの挙動の自然さとプレイビリティの高さでは、いまだこのシリーズを超えるものは現れていない。

移植の際に最も問題になるのは、6ボタンコンパネによる特殊な操作系が必要になることであろう。過去には1ボタンでコマンド化したというAMIGA版の例もあるが、ストⅡは操作感覚



### いろいろあって、目移りしちゃう!?

- |                     |          |
|---------------------|----------|
| 1. コットン             | (前回順位) 2 |
| 2. ネメシス'90改         | 1        |
| 3. 餓狼伝説2            | —        |
| 4. スタークルーザーⅡ        | 4        |
| MATIER Ver.2.0      | —        |
| 6. ぶたさん             | —        |
| 7. ストリートファイターⅡ      | 5        |
| 8. スーパーリアル麻雀PⅡ & PⅢ | 6        |
| 9. SX-WINDOW開発キット   | 8        |
| 10. EG Word         | 7        |

10月号のアンケートハガキのなかから読者の声を集計しました。

さて、1位と2位の順位が逆転しました。先月号で突然1位に登場した「ネメシス'90改」でしたが、今月の集計では「コットン」にわずかに及びませんでした。やはり、興味はとりあえず目先のものに、ということでしょうか。もちろん「ネメシス'90改」への期待が下がったわけではありません。「コットン」の発売は10月号発売の約1週間後でしたので、この号の発売の頃には、もう大半の人は手に入れて遊んでいるこ

とと思います。「ふっか〜つ」の音が耳から離れなくなっている人もいるかもしれません。すでに編集室には「買って満足」の声も届いていますので、メーカーさんには、続編や次回作を期待したいものです。

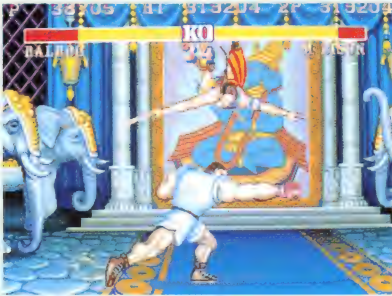
同じく発売間近ということで期待感が高まっているMATIER Ver.2.0。バージョンアップ点などについては、来月号で詳しくご紹介ができると思います。

3、4位の「餓狼伝説2」「スタークルーザーⅡ」の人気は、どちらも前作への高い評価からきているのでしょう。前作に負けない、いやそれを超えるものが期待されます。発売日や価格などは未定で、まだ詳しい情報をお伝えできないのが残念ですが、楽しみに待っていてください。

6〜9位のソフトもそろそろ発売が近づきつつあります。特に7位の「ストリートファイターⅡダッシュ」は根強い移植希望の声のなか、ようやく登場というだけに、来月号の読者の反応が楽しみです。発売は11月26日とのこと。

相次ぐ人気タイトルの発売決定で、財布と相談しながら迷っちゃう人もいるのでは？

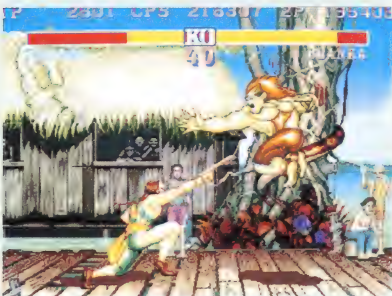




が非常に重視されるゲームなのだ。幸い、X68000版には、スーパーファミコン、メガドライブ用のカプコンスティックファイター(6ボタンジョイスティック)をつなぐアダプタが同梱される予定である。

すでにスーパーファミコン、PCエンジン、メガドライブなどで同シリーズの移植が行われているが、いまなおX68000への移植を望む声は高い。コンシューマ機ではいずれも多少なりともアレンジ版であった。X68000ではやはり忠実移植を望みたい。(S.N.)

X68000用 5"2HD版 12,800円(税別)  
カプコン ☎03(3340)0750

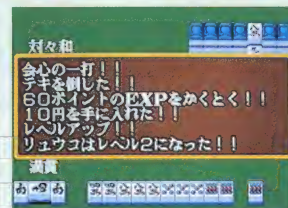


## マージャンクエスト

7月号で紹介して以来、ちょっと情報がとぎれていた「マージャンクエスト」だが、開発は順調に進んでいるようだ。お色気脱衣麻雀にロールプレイングの味付けがされているという、1つで2種類のおしさをもつゲームである。

雀魔王コクシーを倒して世界に平和を取り戻すため、冒険の旅に出る主人公リュウコ。そんな彼女に次々と襲いかかるモンスターたち……(もちろん女の子ね)。戦いを重ねてレベルが上がると、HPとMPが増加する。HPは敵が和了すると減っていき、なくなるとゲームオーバー。使える魔法はMPの量によるので、レベルが高いとつみこみワザがいろいろ選べるようになる。

麻雀を知らない人でも、アドバイスモードな



ら妖精が切る牌を教えてくれるから大丈夫。

発売予定は来春だ。

X68000用  
SPS

5"2HD版 価格未定  
☎0245(45)5777



## スーパーリアル麻雀PⅡ&PⅢ

3人娘と対戦する麻雀ゲーム「スーパーリアル麻雀PⅡ & PⅢ」。

ゲームセンター版の彼女たちは可愛い顔とはうらはらに理不尽なまでに強かったので、さぞかし痛い目にあわされた諸兄も多いはず。なにせコンティニューをしたら、いきなり天和を和了ってゲームオーバー。そんなことは日常茶飯事。いまこそ、そのときの怨みをはらすチャンスがやってきた。しかも今回は、一度勝った分のアニメーションのリプレイ機能がついている。とにかくがんばってショウコ、カスミ、ミキの3人を一度ギャフンといわせるのだ。

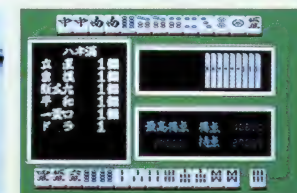
3人のグラフィックはすべてX68000用に描

き直してある。すでに発売されている機種でプレイしたことのある人は比べてみるといいかもしれない。音楽もX68000用のアレンジが行われている。また、X68000版では4人対戦モードにもBGMが入っている。ただ、MIDIに対応していないのがちょっと残念。2人対戦、4人対戦モードのほかに、麻雀を知らない人のための麻雀講座モードがある。

ちなみに2人対戦モードでは、牌をツモるときの手がうに動く。

さあ、みんな秋の夜長に麻雀だ!

X68000用 3.5/5"2HD版 12,800円(税別)  
ピング ☎03(5496)2501



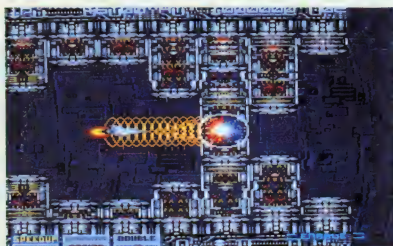
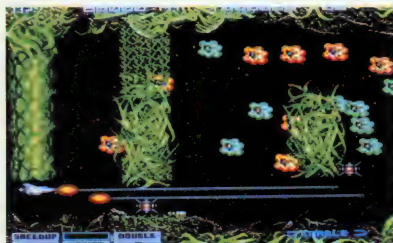


## ネメシス'90改

あの長いあいだの沈黙が嘘のように、開発は着実に進んでいる模様である。ゲーム自体はほとんど完成し、すでに調整の段階に入っている。

人気のグラディウスシリーズのひとつだが、MSX版「グラディウス2」をベースにしたリメイク版。グラフィックの描き直しなどの手が加えられているほか、オリジナルステージもある。これは、完全移植にこだわるのであれば歓迎できるパワーアップといつてよいだろう。

現段階で手もとにあるのは評価版なので、製



品版でどうなるかは不明だが、難易度は非常に高く、骨のあるシューティングゲームとして期待できそうである。発売予定は12月。

X68000用 5"2HD版 価格未定  
SPS ☎0245(45)5777



## 頂劉記

光荣ファンの人、お待ちどうさま。もうすぐ発売の新作は歴史シミュレーションである。

舞台は紀元前200年代の中国。「三国志」の時代より遡ること約5世紀、始皇帝が亡くなり、秦が滅亡したあとの動乱期は2人の武将を中心に展開する。いわゆる「漢楚の戦い」だ。

その武将の名は項羽と劉邦。この2人は対照的な武将である。項羽は武勇の人で自らの力で道を切り開いていくタイプ、対する劉邦は他人の力をうまく引き出すことに優れている人だったという。さて、あなたはどちらの武将でプレイするか？ 劉邦になって史実を再現するもよし、項羽を選んで歴史の可能性を探るもよし。

勝利の鍵を握るのは兵糧の確保と外交政策だが、それだけではない。思わぬところから現れる敵に注意しろ！

X68000用 5"2HD版 12,800円(税別)  
光荣 ☎045(561)6861



## SX-WINDOW開発キットWorkroom SX-68K

待望のSX-WINDOW用開発ツールキット「WorkroomSX-68K」がついに発売される。これは名前とおり、SX-WINDOW上で動作するSXプログラム開発支援環境である。

リソースエディタではリソース化されたデータをボタンやウィンドウ、パターンデータといった要素ごとに専用のエディットウィンドウを開いて設定を変更できる。

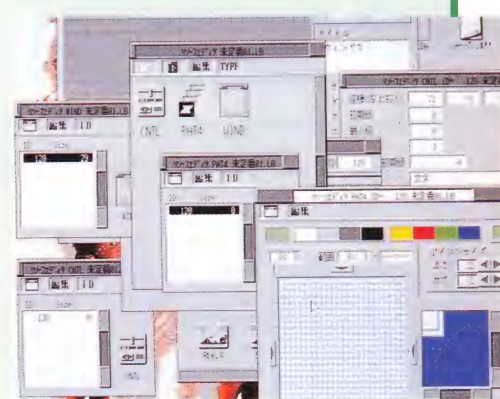
サンプルメイクを使い、プログラムの作成からコンパイルまでの一連の作業をすべてSX-WINDOW上から行うことができる。さらにSX-WINDOWデバッガによって、実行中のプログラムの動作を横で監視しながらデバッグできるな

ど、これまでは難しかったSX-WINDOWアプリケーションのデバッグ作業が効率的に行えるようになった。

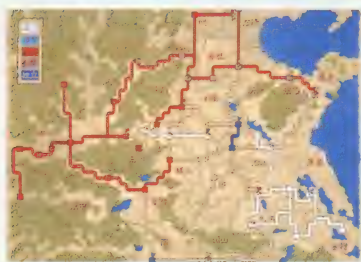
とりえず、これによりようやくC言語用のライブラリにSX-WINDOW ver.2.0に対応したファンクションコールやマネージャなどの仕様が公開されることになる。

もちろんツールだけではなく、各種機能に対応した豊富なサンプルプログラムも付属している。なお、このサンプルの使用にはCコンパイラが必要である。(S.N.)

X68000用 3.5/5"2HD版 価格未定  
シャープ ☎03(3260)1161







## 発売中のソフト

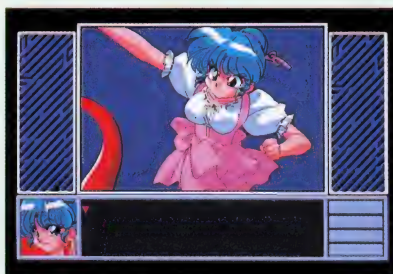
- ★Ultra C & Professional Pack V1.1  
マイクロウェアシステムズ 9/14  
X6800用 3.5/5"2HD版 45,000円(税別)
- ★Technical Tool Kit V2.4.5  
マイクロウェアシステムズ 9/16  
X68030用 3.5/5"2HD版 20,000円(税別)
- ★SX-PhotoGallery 計測技研  
X68000用 3.5/5"2HD版 15,800円(税別)  
(フォトサンプラーCDバンドルセット)  
19,800円(税別)
- ★コットン EAビクター 9/24  
X68000用 5"2HD版 9,800円
- ★宝魔ハンターライム3 ブラザー工業(TAKERU)  
X68000用 3.5/5"2HD版 1,500円(税込)
- ★ダイアット・ヴァークス  
ブラザー工業(TAKERU) 10/15  
X68000用 3.5/5"2HD版 4,800円(税込)
- ★C Compiler PRO-68K NEW KIT シャープ  
X68000用 3.5/5"2HD版 44,800円(税別)

## 新作情報

- ★項劉記 光栄 10/29  
X68000用 5"2HD版 12,800円(税別)

## 宝魔ハンターライム4

毎月1話ずつ発売されるこのシリーズも4回目。今回は、人間界にも慣れてきたライムが、お金を稼ぎながら妖怪退治、という一石二鳥のおいしいお話。すっかり仲よしになったみづきちゃんと一緒にアルバイトするのは、あのアンナミラズ風のお店ののだ。当然コスチュームはアレですよ、アレ。ふたりのミニスカート姿が見られるおいしいお話だ。



画面はPC-9801版です

12話(最終話)までの発売スケジュールも決定した。いろんなところに出没する妖怪たちのおかげで、ライムたちは右へ左へと大忙し。来年の7月まで、まだまだいろんなライムに会えそうだ。

X68000用 3.5/5"2HD版 1,500円(税込)  
ブラザー工業(TAKERU) ☎052(824)2493

### 宝魔ハンターライム 今後の発売予定日

第5話	1993年12月10日
第6話	1994年1月10日
第7話	2月10日
第8話	3月10日
第9話	4月10日
第10話	5月10日
第11話	6月10日
第12話	7月10日



- ★スーパーリアル麻雀PⅡ&PⅢ ビング 10/23  
X68000用 3.5/5"2HD版 12,800円(税別)
- ★ぶたさん 電波新聞社 10/下  
X68000用 5"2HD版 5,900円(税別)
- ★MATIER Ver.2.0 サンワード 10/未  
X68000用 5"2HD版 39,800円(税別)
- ★宝魔ハンターライム4  
ブラザー工業(TAKERU) 11/5  
X68000用 3.5/5"2HD版 1,500円(税込)
- ★ストリートファイターⅡダッシュ カプコン 11/26  
X68000用 5"2HD版 12,800円(税別)
- ★SX-WINDOW 開発キットWorkroom SX-68K シャープ  
X68000用 3.5/5"2HD版 価格未定
- ★宝魔ハンターライム5  
ブラザー工業(TAKERU) 12/10  
X68000用 3.5/5"2HD版 1,500円(税込)
- ★SX-WINDOW 開発キット用サポートツール集 シャープ  
X68000用 3.5/5"2HD版 価格未定
- ★ネメシス'90改 SPS 12/未  
X68000用 5"2HD版 価格未定
- ★卒業〜GRADUATION ブラザー工業(TAKERU)  
X68000用 5"2HD版 価格未定
- ★マージャンクエスト SPS  
X68000用 5"2HD版 価格未定

- ★宝魔ハンターライム6  
ブラザー工業(TAKERU) 1/10  
X68000用 3.5/5"2HD版 1,500円(税込)
- ★宝魔ハンターライム7  
ブラザー工業(TAKERU) 2/10  
X68000用 3.5/5"2HD版 1,500円(税込)
- ★餓狼伝説2 魔法株式会社  
X68000用 5"2HD版 価格未定
- ★ギャラクシーシェイクーズ  
ブラザー工業(TAKERU)  
X68000用 5"2HD版 価格未定
- ★ロボスポーツ イマジニア  
X68000用 5"2HD版 価格未定
- ★Traum 象スタジオ  
X68000用 5"2HD版 価格未定
- ★鮫! 鮫! 鮫! KANEKO  
X68000用 5"2HD版 価格未定
- ★達人 KANEKO  
X68000用 5"2HD版 価格未定
- ★エアバスター KANEKO  
X68000用 5"2HD版 価格未定
- ★サバッシュⅡ ポブコムソフト/グローディア  
X68000用 5"2HD版 価格未定
- ★麻雀悟空・天竺への道 シャノアル  
X68000用 5"2HD版 9,800円(税別)
- ★スタークルーザーⅡ アルシスソフトウェア  
X68000用 5"2HD版 価格未定



## 単純明快爆弾バトル

Shibata Atsushi

柴田 淳

ビデオゲームアンソロジーシリーズの最新作は、ちまちましたぶたさんたちがフィールド上をところ狭しと走り回るバトルゲーム。100匹のぶたの頂点めざし、戦いは繰り広げられます。さあ、その爆弾を投げて投げて投げてくれ！



「ムーンクレスタ/テラクレスタ」から、順調にシリーズを重ねてきた電波新聞社のビデオゲームアンソロジーシリーズ。タイトル選択のシブさもさることながら、その移植度の高さに毎回うならされている読者も多いのではないかな。

ところで、第6弾の「ぶたさん」なのだが、クレジットを入れ、ゲームが始まるまでのしばらくのあいだ、プレイヤーの操るぶたさんが画面上でちらつくのである。

「電波の技術もここまでか」と思った早とちりの読者もいるかもしれないが、そうではない。X68000のスプライト機能は、ひと昔前のゲームなど遥かに凌駕している。つまり、意識的にでなければキャラクターがちらつくことなどありえない。

実は、オリジナルの「ぶたさん」でも、ゲーム開始前のぶたさんは同じようにちらついていたのだ。さりげないことだし、ゲームには一切関係ないようなことなのだが、ここまでこだわってこそ完全移植を謳い文句にできる。これくらいはやって当然の電波なのである。

### 投げる！ よける！ 当てる！

さて、オリジナルの「ぶたさん」は1987年JALECOの作品。なによりストレートなタイトルが目につくゲームで、その名のとおり、ぶたさんが主人公のゲームである。

100匹のぶたさんが、画面上で爆弾バトルを繰り広げる。プレイヤーの目的は、その100匹の頂点に立つことである。ちょうどボ



始める前に、ちゃんとゲームの説明がある

ンバーマンのように、敵を爆弾で倒していくゲームだ、と思っていたいただければ話が早い。

ただし「ぶたさん」は多くの爆弾モノとは違い、フィールドはブロックなどで区切られてはいない。いや、ブロックどころか、ゲームを行うフィールド上には、遮蔽物はなにも置かれていないのである。

爆弾の近くにはぶたさんを動かすと、爆弾を持つことができる。持った爆弾はボタンを押すことで投げられ、またボタンを長く押ししていれば、それだけ遠くまで飛んでいくようになっている。

で、フィールド上には飛んでいく爆弾を遮るものが一切配されていないので、方向さえ定めれば、爆弾は望んだ方向に飛んでいく。また、爆弾は壁に当たれば跳ね返るし、ぶたさんに当たれば爆発をする。

ここまで読めばわかっていただけだと思うが、ぶたさんの基本ルールはひどく単純である。爆弾を、とにかく投げ、よけ、相手を爆発に巻き込む。たったこれだけのこ

とでゲームは進んでいく。

この単純さが、「ぶたさん」のいちばん大きな魅力であるといってい。なんとなくスティックやパッドを握っているだけで、適当なウサ晴らしができる。このことだけでもこのゲームで遊んでみる価値はあると思うのだが、これで魅力が尽きないのが、「ぶたさん」のすごいところなのだ。

### ぶたさんの仕草を見よ！

「ぶたさん」はぶたさんにこだわりとおし、はじめっから終わりまでぶたさんに明け暮れるゲームである。自分でもよくわからない表現だが、要するに、このゲームにはぶたさん以外のキャラクターは出てこない。

プレイヤーの操るぶたさんと弱肉強食のバトルを繰り広げるぶたさんたちには、それぞれ色が割りふってある。それと同時に、ぶたさんには色ごとに独特の性格がつけられている。

ところでここで、このぶたさんたちのように性格づけされたゲームのキャラクターをいくつか思い出してほしい。たいていは、そのキャラクターの動きだとか、または攻撃方法などで特色を出しているはずだ。

しかし、「ぶたさん」の見せる「キャラクターの個性化」へのアプローチはまったく異なっている。まず第一に、ゲームの基本ルールがひどく単純である、ということが、通常の個性化の方法論を入り込ませる余地を著しく狭めている。つまりキャラクターのすることといったら「爆弾を投げ、よけ



X68000用 5"2HD版 5,900円(税別)  
電波新聞社 ☎03(3445)6111



ゴジラスーツは爆風にも耐える



登場するぶたさんはユニークな奴ばかり



る」だけなので、動きや攻撃方法では差をつけようがないのである。

では「ぶたさん」ではどのようにキャラクターの性格づけを行っているのか。ゲームのルールに密接に関わったやりかたが無理なのだから、当然ながら「それ以外の、基本ルールとは関係ない部分」で差をつけるしかない。

各ステージを始める前に、登場するぶたさんについて簡単な解説がある。「はいぶたさん あたまはいいがちよつとにぶい」といった具合にである。で、このメガネをかけたはいぶたさんは何をやるかというと、ゲーム中に何を思ったのか突然本を開き、読書を始める。あまりに本に熱中しているため、爆弾が近づいてきてもぜんぜん気づかない。

そのほか、ゲーム中に不謹慎にも眠り始めるだとか、いきなりなぐりかかってくる、といった具合に、このゲームではルールとは関係ない部分で、キャラクターの個性化が図られているのである。しかも、そのどれもが見ていて楽しいものばかりなのだ。

ところで、この独特なぶたさんたちのアクションは、すべてアニメーションによって再現される。10種類のぶたさんすべてに対して、このようなアニメーションパターンが用意されているのだ。

いまでこそ、ゲームの持つメモリ容量はローエンドのハードディスク並みになってきたが、オリジナルの「ぶたさん」がゲームセンターに出回っていた当時は、使うことのできるメモリ空間はたかが知れていた。そんな状況のなかで「ゲームに登場するキャラクターの性格を、容量を食うアニメーションで再現する」というアプローチは、当時、考えられはしただろうが意識的に避けて通られたのだと思う。

歩くとか投げるとかのぶたさんたちの通常の動きのアニメーションは、基本パターンをパレット機能を使って色を変えることで再現されていたらしいこと、ゲームオーバーの画面に一部ドットが拡大された絵が使われていたことなどを考えると、オリジナルの「ぶたさん」は使用可能なメモリを最大限まで使っていたと推測される。

どちらかというマイナーな部類に入るゲームだとは思いますが、「ぶたさん」はその裏側に制作者たちのこざわりと熱意をかい間見ることができる、隠れた名作ゲームである。マニアックといってしまうとそれまでだが、理屈を抜きにしても、とにかく楽しいゲームだということは万人が認めるところなのではないか。



小休止のぶたたたきゲーム

## ぶたさんの戦略性

話は多少前後するかたちになるが、ここでもういちど「ぶたさん」のルールの話に立ち帰ってみたい。

「ぶたさん」は爆弾を攻撃の中心に据えたゲームなのだが、この爆弾というのは実は時限爆弾である。投げ放つことにより爆弾は点火され、爆弾の表面に張りつけてある数字がカウントダウンする。そして、ゼロになったら爆発する仕組みになっている。

また当然、爆弾は誘爆をする。ここから、次のような戦略が生まれる。

たとえば、とあるぶたさんの通り道に、火のついていない爆弾がまとまっていたとする。そこをめがけて、いまにも爆発しそうな爆弾を投げるとしよう。1個の爆弾で起こすことのできる爆風より、誘爆で起こした爆風は広い範囲に及ぶから、それだけ確実に相手を葬ることができる。

こんなものもある。爆弾を持ち、できるだけぶたさんに近づいて、正面から爆弾を投



ぶたをひっぱたいて日頃のウップンを晴らせ！

げると同時に、ぺこっと伏せる。伏せている状態だと、立っている状態に比べ爆風で死んでしまう範囲が狭いので、相手だけを殺すことができるのだ。

こんなふうには、単純ではあるのだけれど、いろいろな戦略を用いることができるのが「ぶたさん」の奥の深いところだ。そのほか、的を射たパワーアップシステムなど、「ぶたさん」のゲームとして優れている点を数え上げればきりが無い。

1993年3月号の「チェルノブ」のレビューでも同じようなことを書いたが、「ぶたさん」は外見からは想像もできないような深い内容を持ったゲームである。たしかに、かわいらしさは大きな要素のひとつだが、それはあくまでも魅力のうちの一部分である。

特に、普段「俺はシューティングしかやらないぜ」などと豪語しているあなた。この「ぶたさん」をやってみなさい。きっと目からウロコが落ちますよ。落ちたウロコは編集部宛に送っていただければ、僕から素敵なプレゼントが……。



ゲームオーバーの画面。相合傘が泣ける



意味不明。どうしてぶたさんが宇宙に？

## 電波さんありがとう！

オリジナルが出回っていた当時、「ぶたさん」は僕がゲームセンターに行くとき必ずコインを入れるゲームだった。しかし、商業的にはあまり受け入れられず、不遇に終わったゲームだと記憶している。どうしてこんなに面白いゲームが、多くの人にプレーしてもらえなかったのか不思議でしょうがない。間違っているのは、僕か、それとも大衆か？

しかし、電波新聞社が「ぶたさん」を移植すると聞いて、僕はあんまりうれしいので小躍りしてしまった。

「チェルノブ」のレビューで「ぶたさん」を移植してくださいみたいなことを書いたし、半分期待を込めていつか移植されるだろうとは思っていたのだが。電波さん、これからも僕たちによいゲームを届けてやってください。

総合評価	0	5	10
かわいい			★★★★★★★★
ぶたさんたち			★★★★★★★★
手軽さ			★★★★★★★★
軽快な音楽			★★★★★★★★
移植度			★★★★★★★★



# 獣と化せばパワーアップ

Sudou Yoshimasa

須藤 芳政

戦いの舞台は、ファンタジー世界「ダイアット・ヴァークス」。人類と亜人類（獣人）が共存し、法王と守護精霊たちに統治された平和な世界が一転して……。現実にもちょっぴり疲れたそこのあなた、世界を救ってみませんか？



シミュレーションゲームといえば思い出すことがある。

高校時代、我が校には「シミュレーション同好会」なる集団（というほどの人数を獲得していたのかは、かなり怪しい）が存在し、その会長は恐れられていた。別に空手家やカポエラ使いだったわけではなく、次々と繰り出される常人の理解から脱線転覆した奇怪な行動が「恐れられていた」のだ。

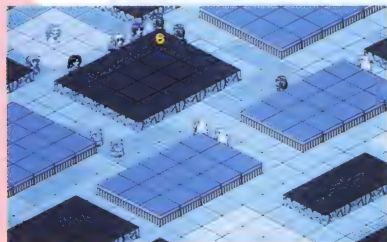
そのような理由から、私はシミュレーションゲームとは常人がプレイするものではないという考えが頭から離れなかった。シミュレーションゲームに関してはまったくの初心者で「信長の野望」から「ゴム長のヤボ用」を連想してしまうし、六角形がたくさん目の前にあってもロイヤルゼリーしか頭に浮かばない。

そんな私が今回プレイしたのがこのダイアット・ヴァークス。ストーリー進行に沿って戦闘を勝ち抜き、目的を達成する「サイバーファンタジーシミュレーションゲーム」だ。横文字に弱い私には何のことかわからないが、サイバーでファンタジーなシミュレーションゲームということだろう、おそらく……。

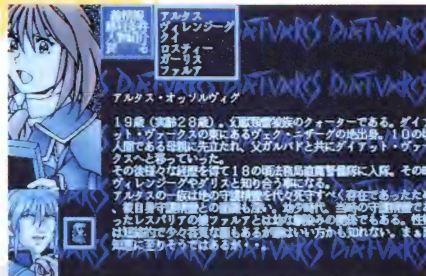
別に「あたしい、食事制限しなくちゃいけないしいー」とは関係ない。それはダイエットである。

## 君たちは人間ではないのか？

舞台は人類と獣人が共存する世界。獣人のなかで普段人間の容姿をしている



X68000用 3.5/5"2HD版 4,800円（税込）  
ブラザー工業（TAKERU） ☎052(824)2493



幻獣類雷狼族のクォーターのアルタス

者は、獣化すると狼、豹、ドラゴンに変身するのだ。ドラゴンが徘徊しているわりには、道路が舗装でビルディングがそびえているという設定は面白い。

ここで気づいた点がいくつかある。まず、この物語の中心的人物である獣人アルタスを筆頭とする登場人物の男たちはなぜか長髪だ。この世界では長髪が普通なのである。電車に乗ってもチビッ子から「あのお兄ちゃん女みたいー」という声が発せられることもないし、引越しのアルバイト募集で「長髪不可」という制限にタメ息をつくこともない、「長髪だよ全員集合」といったところか。

次に人類と獣人の共存であるが、果たして可能なのだろうか？ 友達になれるのか？ 私はカレーを食べてラモス選手に変身してしまうまさお君でさえ受け入れることができない。

そして獣化後の服の行方は……。

ここまで書いて思ったが、私は非常に余談が多い。

## 守護精霊を守るのだ

さて、法王が最高位として治めているこの世界は8人の守護精霊によって自然のバランスが保たれていた。しかし、法王の予言によりこの世界を恐怖に陥れる危険な存在であるとされた守護精霊は、法王の差し向けた部下に次々と殺害されてゆく。すでに5人の守護精霊が命を落としており、この事態に座談会（お茶、菓子類は見当たら

なかったが）を開く獣人の若人連中がいた。

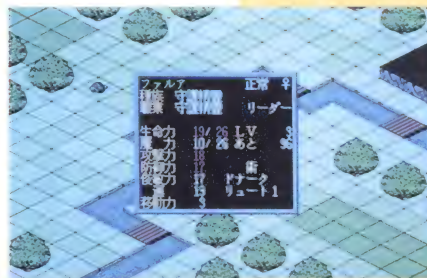
そのうちの1人、アルタスが幼なじみであった守護精霊ファルア（本当はファリネイシア・リカル・ラ・ジェルバートンというらしい）の無事を願っていたとき。そこへファルアが久しぶりの再会にもかかわらず土産のひとつも持たずにひょっこりやってくる。続いてファルアを追って、警察隊が登場！ なるほど、ファルアは追われていたのだ。警察隊は、ファルアを差し出さなければお前たちも皆殺しだという。どうする!? ここでファルアを見殺しにしてしまっただけでゲームが終わってしまうではないか！ ゲームが終わってすることといったら寝ることぐらいしかできない。ええい、やっておしまい！ かくして戦いは始まった。

## おまわりさんにアタック！

まず警察隊と戦うのだが、ティリア（千里眼を持つ彼女は戦いには直接参加しないが獣人らしい、獣化しないのは先に述べた服の行方と関係があるのか？）に敵の情報や、勝利条件などを教えてもらおう。勝利条件は「敵のリーダーを倒す」「敵を全滅させる」など、場面によって異なるので必ずチェックしておくこと。

次に、戦闘に参加する仲間を選び、そのなかでリーダーを1人決める。リーダーが倒された時点で戦闘は終了なので、打たれ強そうなヤツを選んでおいたほうが良いだろう。

いよいよ戦闘開始。画面は戦闘場面を斜



レベルが上がった



めから見下ろした（クォータービューというものらしい）状態で、各キャラクターの移動可能範囲で移動、そして敵への攻撃を行う。

メンバーのうち、アルタス、ヴィレンジークは通常攻撃でまあまあ敵と対等に戦えるが、ダリスは攻撃力が弱い。「なんだよコイツ使えねーな」と

思いながらよく見ると、彼は強力な魔法が使えるし、獣化すると強いのだ！でも、まだ獣化はしない。一度獣化すると次からの1または2ステージ（キャラクターにより異なる）を終了するまで獣化できなくなる。獣化するとヒットポイントが全快するというメリットも考慮すると、ピンチに陥ってから獣化しても遅くはないであろう。

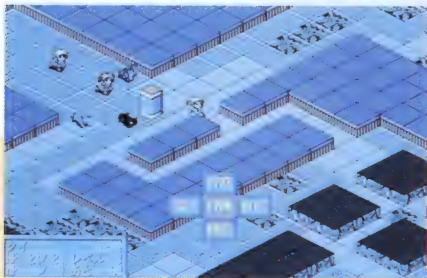
なんとかいろいろやっているうちに勝ってしまった。警察隊との戦闘はこれからゲームを進めるにあたってのいい練習ステージになる。

## ロバ参上！

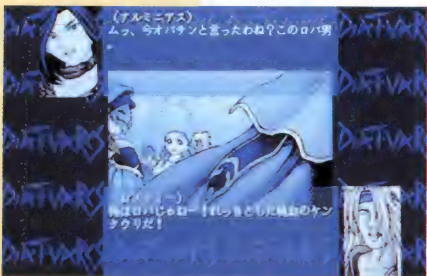
無事に警察隊の包囲を突破、もう1人の守護精霊ミスティアに会うために、彼女がかくまわれているというケンタウリ（上半身人間で下半身が馬の種族）の集落へと向かった。

とそのとき、目の前に立ちはだかる2頭のロバ！ではなくてケンタウリだ！

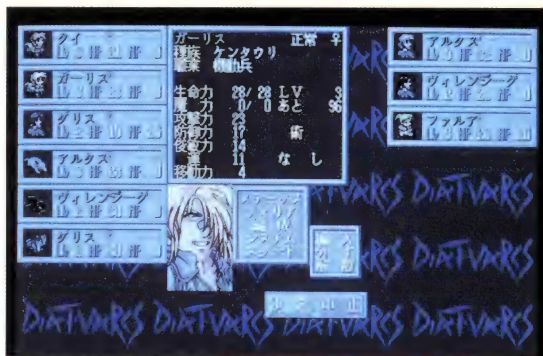
2人はファルアが守護精霊であることを信用していないらしい。自分が守護精霊で



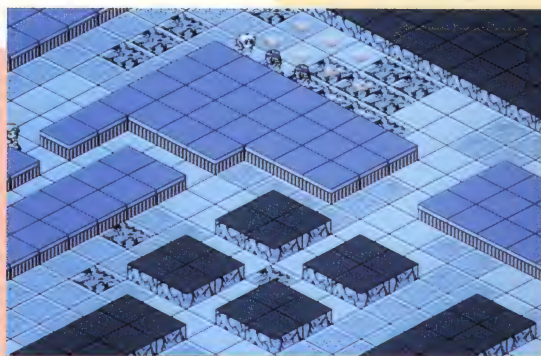
仲よく並んだロバ姉弟を攻撃



お姉さまに失礼な役たつたロバ男



戦いに参加する仲間を選択



近距離で爆発すると自分も巻き添えに

あることを証明するためにたった1人でケンタウリたちと戦うことになったファルア！

「えーい！ ポカポカ！」

勇敢に殴りかかったファルアであったが、あっという間にボコボコにされて敗北。ここは魔法を使わなければ勝てそうもないので、次は魔法を使ってみる。すると地震が起こってあっという間に勝ってしまった。さすが「地」の守護精霊だ。

しかし、地面の揺れだけで人を生命の危険にまで陥れることは、果たして可能なのだろうか？ だとしたら地震体験車で地震を体験した人はその直後全員病院送りだ。

## オーッホッホッホ！

ケンタウリの集落へ到着するが、ミスティアは不在。探しに行ったところ、ミスティアと彼女の護衛についていったケンタウリ族のロスティーが敵に囲まれていた。

敵の大將は「オーッホッホッホ！」と高らかに笑うアルミニウスお姉さま。「オーッホッホッホ！」と笑っても、彼女は心のスキマを埋めてくれるセールスマンではないことに注意しよう。

ここで戦闘に入ると「おや？」と思うはずだ。そう、いままでより戦いのフィールドが拡大されている。戦闘開始直後では敵がどの地点に存在、移動しているのか把握できない。

それにこのステージからは、いままでの戦い方で勝つことは難しくなってくる。戦略を練ることが必要だ。

このステージは最終的にアルミニウスお姉さまを倒せばよいのだから、余計なザコとの戦いは極力避けたい。どうすればよいのかというと「オトリ作戦」である。ケンタウリ族のロスティーとクイは本当に「つかえねー！」と心の底から叫んでしまうほど使えないヤツラなので、わざといったん敵の近くまで接近して、近づいてきたら一定の距離を保ちつつ敵を誘導しながら味方

の集団から離れ、そのスキにほかのメンバーはアルミニウスを倒せばいい。

ケンタウリ族のガーリスは直接攻撃よりも、相手と少し距離を置いて「間接攻撃」を行ったほうがいい。これは爆発が起こって広範囲の敵に攻撃できるものだが、ダメージを食らうのは敵味方関係なしなので注意が必要。でも、ロスティーやクイが巻き添え食らったって気にしない気にしない（クイはガーリスの弟だけど……）。

## まだまだ先は長い

このゲームが本当に面白くなるのは、これから先だと思う。このテのゲームが苦手な私はなかなか進めない。しかし、やめられない。この「やめられない」状態になってしまうのがシミュレーションゲームの魅力なのだろうか？

1つのステージをクリアするまでにかかる時間が結構長いので、負けてしまったときは精神的ダメージが大きすぎて、本当に「うおー！ 人生とはー！」と叫びなくなった。セーブを忘れていたときは、もう何もかも捨てて旅に出ようかと思ってしまったほどだが、ステージの合間に見られるメンバー同士の会話は達成感を増幅し、次のステージへの期待を持たせてくれた。

ぜひ法王と対決してみたいものだ。

しかし、私はまさにお君がラモス選手になるのはやっぱり認めない。

## ジョイパッドで混乱します

クォータービューなのでジョイパッドでプレイするとどうしても方向を斜めに入力してしまうのだが、入力を受けつけるのはタテ、ヨコのみ。それではパッドを斜めに傾けてフィールドの角度と同じにしてはどうかとやってみたが、今度は攻撃選択の角度と合わなくなってストレスが溜まってしまった。

総合評価

操作性

絵

音





## AFTER REVIEW

8月号で紹介したバトルゲーム「ロボット  
コンストラクションR.C.」は、パーツを組み  
合わせて自作したロボットの動きをプログ  
ラミングする。そのロボット同士を戦わせ、  
最強のロボット作りを目指すのだ。



### ロボットコンストラクション R.C.

- ▶「餓狼伝説」を末梢神経系のゲームとする  
と、「R.C.」は中枢神経系のゲームだ！ は  
かのロボットとの「知恵くらべ」が楽しい  
ゾ！ 池田 譲太(25)大阪府
- ▶限られた表現のなかで行動をデザインす  
る。むしろ人生よりも露骨に。  
中島 民哉(23)埼玉県
- ▶ひさしぶりの徹夜でした。  
相沢 栄樹(26)東京都
- ▶キャラクターが小さいが、なんだか愛着  
がわく。 横田 大介(18)北海道
- ▶なかなか勝てないけど、少しずつ強くな  
る姿がいい。 星沢 厚志(23)熊本県
- ▶いままでにあまりないタイプのゲームで、

とても楽しめた。 渡辺 現(19)大阪府  
▶気長に末長く遊べる。各人が作ったロボ  
ットと戦うのも面白い。

- 後藤 幸夫(32)宮城県
- ▶ロボットが思いどおりに動いたときは最  
高。そこまでの苦労があるけど。  
光石 和広(20)神奈川県
- ▶会社で昼休みに遊べる。  
岡邑 信吾(18)大阪府
- ▶ルーチンを改良してくのが楽しい。  
澤田 裕史(18)神奈川県
- ▶頭をよく使うゲーム！  
小野寺 学(23)北海道
- ▶プログラミングの「はがゆさ」が戦闘意  
欲を燃やす。 大塚 啓治(32)兵庫県
- ▶パソコン通信でみんなで遊べる。  
嶋 真一(28)大阪府

### バトル大会も開催

このゲームのパッケージにはいろいろな  
タイプのロボットデータが入っているの  
で、もちろんひとりでも遊ぶことができる。  
しかし、自分のロボットを作ったら、たく  
さんの相手と戦ってみたいのが人情とい  
うものだ。そこで、発売元のエレクトリ  
ックシープでは、ユーザーにさらに楽し  
んでもらうための企画として、ロボットバ  
トル大会を開催している。参加方法は2  
通りあり、ひとつは郵送、もうひとつ  
はパソコン通信によるものである。後  
者のほうは、NIFTY-Serve上ですでに  
2回行われた。以下に、その模様を紹  
介しよう。

ちなみに、左ページの写真は両大会に  
参加したロボットたちをOh!X編集室で  
戦わせたものである。したがって、大  
会での戦いとは必ずしも一致はしてい  
ない。

#### ロボットバトル大会on NIFTY 第1回

参加ロボット：40体

●Oh!Xより2体参加

柴田 淳 (HOT-SHOT)

高橋 (KONAIDE)

大会の形式は、まず予選として総当り  
戦での勝ち、負け、引き分けの数により  
ポイントを計算しました。決勝は上位8  
体を残し、メニューからオートトーナ  
メントを遊ぶという方法で行われました。

全体の印象は、待ちロボットが多く  
て、強いときは強いが、とにかく待ち  
ロボット同士のDRAW GAMEが多かつ  
たとのこと。結果は、Oh!XではKONA  
IDE(高橋)が予

選7位で決勝進出を果たしましたが、惜  
しくも敗退。初めての大会ということ  
であってか、主催者エレクトリックシー  
プの酒井智己氏のHOIYERが優勝。酒  
井氏のコメントは、『「作者のくせに  
予選落ちかよおお」とかいわれると  
怖いので、予選は通過したいと思っ  
ていましたが……』とのことでした。

HOIYERは優勝ロボットとして、次  
回大会に自動的に出場することになり  
ます。

#### ロボットバトル大会on NIFTY 第2回

参加ロボット：61体

●Oh!Xより1体参加

柴田 淳 (Y-JACKET)

参加者が増えたため、予選はA、Bの  
2リーグに分けて行われました。

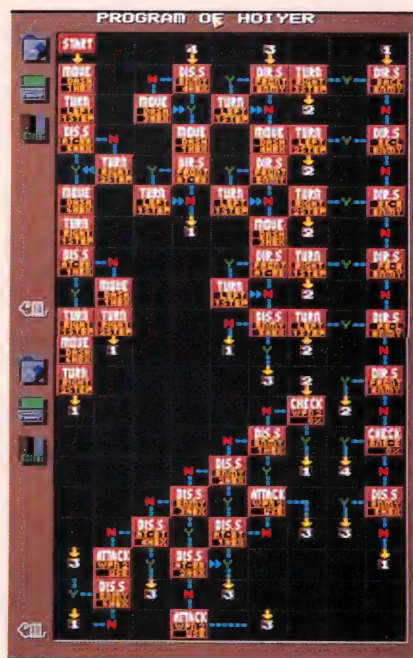
今回参加のロボットのスペックを見  
てみると、火力重視(威力がある、ある  
いは火をつける)タイプが多いよう  
でした。第1回の経験からか、皆さん  
今度は「勝てる」ハードウェアを選  
んでいるという感じです。

足は、全体の77%(47体)が2足  
です。次はいきなり減って6足の5  
体。あとはそれぞれ3体ずついます。  
武器の一番人気はやはりSPARK。21  
体のロボットが装備しています。次  
が、FLAMEとLASERで、それぞれ16  
体。ほかにはSHIELDやMINEが多  
いほうです。ATOMICは6体に減り  
ました。いちかばちかのギャンブルを  
恐れたのでしょうか。

優勝は、ご本人の「優勝宣言」ど  
おり、7COLOR-2(Binaryさん)でし  
た。



## 第1回大会の優勝ロボット 「HOIYER」のプログラム



第1回大会決勝：HOIYER vs L-FIRE

HOIYER(コンストラクター：エレクトリックシープの酒井智己さん。)とL-FIRE(PuPiさん)。L-FIREはいったん決勝進出圏外に落ちたものの再浮上。予選のランキングは4位と8位で、過去の対戦成績は2勝1敗0分でHOIYERが1勝分りードしていた。画面左上にのいるのがL-FIRE。編集室でのこの対戦もHOIYERが勝利をおさめた。



左上 CHASPID(千秋さん)：6角形の陣を張る。一生懸命花壇を作ってるみたいでスタッフにウケたそうだ。

右下 M-MASTER(TENTENさん)：完璧な地雷配置で、その後の戦い方も見事。が、順位はなぜか37位に終わってしまった。たまに硬直するので、論理ミスがあるのかも、とのこと。

FUJI-N3(佐藤潤一さん) vs NIGHTMAR(神威さん)

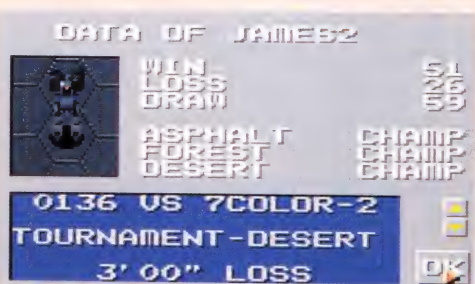


ZABORGER(内野研究所さん) vs DANGAI-8(ダンガイ男さん)

## 第2回大会の決勝進出口ロボット



7 COLOR-2(Binaryさん)



JAMES2(KAZN'CHIさん)



柴田氏のロボット2体。第1回のHOT-SHOTは6足タイプでずんぐりと可愛い。第2回のY-JACKETは2足タイプで見た目はスマートだ。それにしてもカラーリングが同じってどこに柴田氏の好みが見えているのだろうか。ここではHOT-SHOTの勝ち。



Oh!Xスタッフ対戦。第1回大会で予選7位のKONAIDE(高橋)に挑むは、Y-JACKET(柴田)。結果は1分54秒を残すところでKONAIDEの勝ち。決勝進出の面目躍如か。しかし、2回連続出場経験値が増えているだろう柴田氏の第3回出場ロボットはあなどれないぞ(たぶん)。



第2回大会の決勝は7COLOR-2 vs JAMES2。編集室の対戦でも、やはり勝者は7COLOR-2だった。

## 今後のバトル大会の予定

- 郵送による参加の全国大会(9月末締切済み)
- R.C.大会 on NIFTY 第3回(10月末締切)
- NIFTY-ServeのFSHARP2「ゲームの部屋(mes 11)」にはユーザーの作ったロボットが多数アップロードされているので、興味のある人はのぞいてみるといいだろう。



TREND  
ANALYSIS

1993年10月号のハガキ集計ベスト10 最近買って気に入ったソフトは?

POINT	タイトル	発売元	発売日
189	悪魔城ドラキュラ	コナミ	'93/7/23
74	クレイジークライマー/ クレイジークライマー2	電波新聞社	'93/8/27
66	餓狼伝説	魔法株式会社	'93/7/23
34	リブルラブル	電波新聞社	'93/6/25
29	エトワールプリンセス	エグザクト	'93/3/26
25	SX-WINDOW ver.3.0	シャープ	'93/3/30
25	The World of X68000	電波新聞社	'93/8/20
20	ロボットコンストラクションR.C.	エレクトリックシーブ	'93/7/30
12	コットン	EAビクター	'93/9/24
12	Easydraw SX-68K	シャープ	'93/8/5

(無作為抽出した1000通のハガキを集計)



やはり強し！先月に続き圧倒的なポイントを獲得している「悪魔城ドラキュラ」である。ゲームそのものだけでなく、グラフィックやサウンドなどのあらゆる部分についてとことん作り込んである作品だけに、添えられている読者のコメントにも熱がこもっている。アンケートハガキとは別に、攻略の図解なども編集部に寄せられているので、それらについては機会をみて紹介していきたい。もともとユーザーの支持が非常に高かったコナミであるが、この作品でさらに「信者」を増やしたといってよいだろう。

2位には「クレイジークライマー/クレイジークライマー2」が登場。これのために「リブルラブル」を購入したという人もいう話だが、前評判にたがわず人気は上々だ。ひたすら登る、という単純なことながら、そこにさまざまな要素がからめられている面白さが魅力のゲームである。

3～6位は先月号の2～5位と同じ並びになっている。ポイント数そのものは先月号とあまり大差ないので、人気が落ちてきたということではない。「クレイジークライマー/クレイジークライマー2」に押し下げられた形である。

先月号より登場の「The World of X68000」「ロボットコンストラクションR.C.」も安定した人気を保っている。

9位には、発売が決定される以前より移植希望の聲が高かった「コットン」が初登場である。集計時点ではまだ発売されたばかりだが、さっそくハガキが寄せられた。すべり出し好調といったところで、来月号のポイントが気にかかる。なんといっても、可愛いキャラクターとそれに反する結構シビアな難易度、というのがゲームマニアの心をくすぐるのだろう。

そしてもうひとつ、これも発売間もない「Easydraw SX-68K」がランキング入り。8月号で紹介済みだが、今月号の75ページでも製品版に基づいてレビューしているので、興味のある方は参考にしてほしい。6位の「SX-WINDOW ver.3.0」の人気とあわせて考えると、X68000の使い方に関して変化の兆しがあるのかもしれない。「SX文化」がどうなるかは、今後のSX関連のツールの発売いかんにもよるだろう。

ここ数カ月は人気タイトルの発売が相次いでいるので、全体的にポイント数が増えてきた。コメントを見ても満足度は高いようである。他機種に比べてソフトのタイトル数は少ないが、厳選された質の高いソフトばかりならばX68000ユーザーにとっては幸せなことだといえるだろう。これから年末に向けても、次々とビッグタイトルの発売が予定されている。

さあ、「戦国時代」に突入か!?



【特集】

# SLASHの活用

ひと昔前の3Dグラフィックの流れは、よりリアリスティックな画像を目指していました。レイトレーシングなどによる光の反射や屈折から始まり、淡い陰影や炎の揺らめき……。

リアルタイムレンダリングを行うハードウェアが開発されるようになってからは、皆さんご存じの「バーチャルリアリティ」という流れが現れてきています。まだまだ言葉の持つイメージだけが先行している概念ですが、試行錯誤を繰り返しながら着実にかたちのあるものへと成長しつつあります。3Dグラフィックのリアルタイム性や、インタラクティブであることの価値が見直されてきているといってもいいでしょう。

これは「画像を作る」というところから一步進んだ技術です。むしろ生成される画像をいかに制御するかというシステムが重要なのだといえます。

画像生成が目的であったCGと画像生成が手段であるCG。私たちはSLASHというシステムを手に入れました。これをいかに使っていくかということが重要になってきます。使いこなすまでのレベルに到達するにはたくさんの段階を経なければなりません。

まず使い方からの把握から始める必要があります。サポートツールももっとも必要です。しかし待っているだけでは問題はなにも解決しません。

一緒に最初の一步を踏み出してみましょう。

## CONTENTS

3D処理の可能性を探る……………	中野修一
モデルの修正と拡張……………	菊地 功
回転体生成プログラム……………	田村健人
ポリゴンソートフィルタ関数SortPoly() ……	丹 明彦
とりあえず三角錐を回してみる……………	山田純二



# SLASHと関連ツールから見た 3D処理の可能性を探る

Nakano Shuichi 中野 修一

多くの人が夢見ていたポリゴナイザ。ではポリゴナイザがあると、どんなことができるようになるのでしょうか。ここではSLASHが開くべき世界を探っていきましょう。

横内君が命を削って作ったポリゴナイザSLASHは反響も大きいようです。なかには、これで限界が見えてしまったとなぜか落胆する人もいたようですが、発展途上のシステムですし、まだ使いこなすといえるほど使った例もありません。モデリングデータの切り替えや画面クリアの最適化（ちづついても高速クリアモードをつけるべきか？ 全画面クリアのほうが速いか？）など検討すべき問題は山積みです。

ということで、読者の皆さんにプロジェクトへの協力を要請したわけですが、さっそく三角関数の積を和に展開する方法が送られてきました。なにに、これによるメモリは1ポイントあたり400クロックですか……。1ポリゴンあたり4クロック落ちるから云々でPC相対のデータアクセスに未練を残していたSLASHからすればまさに福音といえるものです。

そのほか、自作システムでのアルゴリズムなどを送ってくださる方もいました。まだまだ改善の余地があることは証明されたわけですから、引き続きご協力をお願いいたします。

\* \* \*

さて、実用的な速度で動くポリゴナイザがあるということによってどのようなことが可能になるのでしょうか。

とりあえず3Dのゲームが作れます（当然）。さらに3面図などでは感覚的にわかりにくい3Dオブジェクト用のエディタなどでも、インタラクティブな操作環境が実現できるということになります。3Dグラフィック全般について応用ができます。そして3Dドローイングツールや3Dデータベース、プレゼンテーションやサイエンティフィックビジュアライゼーションのようなものにわかに現実味を帯びてきます。ゲーム用に作成されるものはそれだけ高性能が要求されますから、ゲームという目的のみならず、さまざまなアプリケーションに応用が

きくのも当然かもしれません。

## まずはモデリングから

SLASHというのはライブラリのかたちにとまめられています。さらに開発者の意思にはやや反するものの、C言語から扱えるような関数群も用意されています。これはできるだけ多くの人に使ってほしいということの意味しています。

ライブラリというものは使うためにあります。SION IVが現れるまでじっと待っているようではいけません。なにか、もの凄く難しそうなものだと思って敬遠している人もいるかもしれませんが、難しい部分はほとんどすでに処理されています。異様に迅速に開発ツールも揃えられました。あとは指定された使い方をすれば、SLASHの機能はあなたのものになります。

SLASHでは、すでに形状エディタが用意されているので、少なくともMAGICのときよりはアプリケーションの作りやすい環境が揃っています。「バグが多くて……」と作者はしていますが、それなりに使いものになるモデラだと思えます。これがなければSION IVのデモはとうていできなかったことでしょう。

私もちょいちょいと使ってみました。当初予想していたよりも簡単に扱えました。とりあえずF16Aを作り、その途中パーツからRAFALEを作り、操作に慣れてきたので勢いにまかせて一気にフォントデータを作り……。は、いいのですが、当初構想にあった3Dテキストエディタはやはり重くなりそうですね……。

\* \* \*

SLASHはプログラムを作るには敷居が高いシステムかもしれませんが、開発環境のない人でもモデラだけで結構遊ぶことができます。ポリゴンシステムによる開発はモデリングに負うところが大きいので、皆

さんも面白いデータができましたらぜひ送ってください（オブジェクトライブラリができるというのですが）。

さて、しばらく触っているとモデリングのコツというものもわかってくると思います。特に面定義の順番はもっとも重要な部分です。先月号の付録に収録されたF16Aなどは、このあたりがまだ煮詰まっていない部分もあります。なにしろ、モデラにカレントポリゴンの優先順位をひとつずつ移動させる機能がついたのはマスターアップの直前でしたので。

丹氏がSortPoly()関数によって面順序の自動処理を目指していますが、順序関係が破綻するデータではうまく対応できません。現状ではこういったものはやはり手作業で処理する必要があります。

裏は描かない

定義順に描く

といった簡単な規則で大丈夫なのかと思う人も多いと思いますが、面をうまく組み合わせれば意外と複雑なものまでちゃんと表示できるものです。しかし、多くのもので破綻が生じることがあるのも事実です。

そこで、順序関係に破綻を生じる場合のコツというか、基本的な戦略は「影響を最小限に留める」ことです。捨てる部分を明確にし、矛盾が生じてでもできるだけ気にならないようにします。破綻の原因は前後関係の循環ですから、ある特定の角度を捨てることでほかの部分の救うことができます。自動車なら発生する矛盾を真下から見たときに集約するとかいった対処が考えられます。問題が集中していれば、マクロソートを使って完全に破綻を除去するのも簡単になります。

いうまでもなく、よいモデリングというのは、より少ないポリゴンでより整ったものを作り上げるということです。

重要なのは単純化することです。全体的なイメージさえ捉えていれば細部は無視し



たり、大幅なアレンジを加えたほうがよいことだって往々にしてあります。資料とにらめっこして作ったRAFALEよりも記憶だけで作ったF16Aのほうがそれっぽいのも気のせいではないでしょう。

モデラは、今回行われた拡張で対称機能がちゃんと使えるようになりました。世の中の物体すべてというわけではありませんが、ゲームなんかで使うもののたいいは左右対称ですからこの機能は重宝すると思います。さて、この機能を使うときのコツですが、たとえば右半分をすべて作っておもむろに対称化したりしてはいけません。面の定義順番というものがありますから、そのあたりをよく考えながらやったほうが面順番の修正が楽になります。

## 色の指定

先月号では詳しく記載されていませんでしたが、SLASHではとりあえず使えるものということで基本16色というのが設定されています。std00~15のようにカラーコードでも参照できますが、それぞれ、

```
std_black
std_darkgray
std_darkblue
std_blue
:
```

のようにラベルがつけられています。

これらはTXEDで作成されたものですが、32段階のうち最高度付近をハイライトに、最低度付近を影として処理しています。ですからどのような色でもだいたい均等に陰影づけがされるわけですが、これは暫定のもので、実際の使用時には背景色に対して調整しなおされることが必要です。コメント部にだいたいの色の位置が書いてあるので参考にしてください。

また、特殊な用途ではもっと急速な陰影変化が要求されることもあるかもしれません。そういった場合は左右クリックによる範囲外指定によって対応します。環境光強度をマイナスに設定したり、明るさを上限以上に持っていったりすることで、ある程度の調整が可能です。なお、この機能はコンパイル前のBASICプログラムでは使用できません。また、多少ゴミが入ることがありますので必要があればエディタで修正してください。

面の色彩はTXEDで作成されるような色の配列で決まります。光源に対する角度によってあらかじめ色を決めているだけで、もっととサイケな色彩にすることも

簡単です。色相を回してカクテル光線ぼくしたり、ハイライト部分に色彩を乗せて色つきの光源にしてみることもできます。幼い頃、赤、青、黄色の3色の光線をあわせて黒い光線を作るというロボット怪獣を見て（大空魔竜ガイキングだっけ？）斬新さに心打たれたことがありますが、このシステムなら黒い光源も簡単です。

光源は物体単位で設定できますから、背後の爆発などで瞬間的に逆光にしてみたり、点光源のような感じを出したりすることが考えられます。かなり自由度の高いシステムなのでいろいろな技が使えるそうです。

## マッピングの可能性

色指定用に出力されたファイルを見てもわかりますが、システムの仕様上の問題から1階調につきカラーコードを2つずつ指定しなくてはなりません。違う色を指定しても問題はないのですが、1ドットごとに違う色になってしまいます。

SION IVデモ版の敵弾では故意にこれを変えた色が使用されています。ときどきメッシュ状になっているのが確認できると思います。基本的にラスター単位の処理ですのエッジの状態によっては縦縞になることもあります。常にメッシュにするように改造することも難しくないとは思いますが、SLASHは65536色がちゃんと使えるポリゴナイザですから必要はないだろうということでサポートはされていません。

ポリゴンの塗り潰しをタイルパターンにすることは理論上可能です。横30ドットまでなら楽勝でしょう。メッシュは使いようによってはテクスチャマッピングっぽい表現も可能なのかもしれませんが……。

SLASH開発時の合い言葉は「リアルタイムでD6GAの画質」というものでしたので、もう一歩進んで（無謀なことと知りつつも）、リアルタイムテクスチャマッピングの可能性について考えてみましょう。

まず、マッピングというのはどれくらい重くなるものなのでしょうか。

現状ではMOVEMを使っているのでも1ドットあたり4.5クロックで描画できます。これをMOVEにすると1ドットあたり6クロックになってしまいます。これだけならたいしたことはありませんが、1ドットごとに対応する色彩を持ってこなければならぬのでうまくやっても10倍の時間は覚悟したほうがいかもしれません。

要するに、もの凄く重いわけですが、X68030専用と考えるとどうでしょうか。問

題は計算量ですからCPUパワーの違いがそのまま出てきます。

無駄な描画は命取りですので画面の手前から描いていくアルゴリズムが有効になりそうです。ラスト抜きはほぼ確実に2倍の高速化を実現してくれるでしょう。

つい最近までD6GA CGAシステムは2次元変形だけでマッピングを行っていたわけですが（結構大胆な気もする）、この方法でもアポロやジャンボルガーの画質が得られていたのですから、まんざら捨てたものではありません。計算量はかなり少なくなります（まだ重いけど）。やはりSLASHではマッピングデータは輝度別に32種類展開しておくべきでしょうか？

このように考えていくと、X68030によるマッピングポリゴンはまるっきり不可能というわけではないかもしれません。しかしこれにスムーズシェーディングが加わると重さも倍増してきます。輝度の直線補間間に合わせるとしても……ちょっと重すぎますか……。

## 次世代を目指して

おそらく、マッピングポリゴンなどは楽々こなしてしまう次世代ゲーム機も2年以内に普及することでしょう。確実に地盤を固めつつあるエレクトロニックアーツ&松下の3DO、まだ見ぬ強豪といった感じのアタリ&IBMのJAGGER、いつになるかはわかりませんが任天堂&SGIのニューマシンの顔ぶれが並びます。それらはハードウェアによる処理ですから、現状のパソコンでは到底太刀打ちできません。

それでもパソコンによるポリゴン処理に魅力があるのは、自分の手でなにかを作り出すことができるということにあります。実際にどれだけ応用するかということよりも可能性が存在するということが重要なかもしれません。

ビデオの画質にまったくかなわないのにハードディスク一杯にQuickTimeムービーをため込む人を笑うのはたやすいのですが、そこから先に広がる世界が見えていれば笑ってもいられないでしょう。このあたりはパソコンの本質にもつながるところがあるような気がします。

ポリゴンという表現方法自体がまだまだ一般的ではない状況ですから、すべきことは山ほどあります。次世代にはパソコンだってもっと進化することでしょうし。

ただ、個人的にはワイヤーフレームの出す「味」がもっとも好きなのですが……。



# 特殊機能のデバッグ モデラの修正と拡張

Kikuchi Isao 菊地 功

10月号の付録ディスクに収録されていたSLASH用の簡易モデラをバージョンアップします。これまで不安定だったり、問題があった部分がほとんど解消されました。指定の面倒だったポリゴン消去も簡単になっています。

先月号の付録ディスクに収録されたSLASHモデラですが、皆さんもう使ってみられましたでしょうか。暫定版ということでバグが残っていると注意しておきましたが、一般的な機能しか使いものにならなかったのが今回はそのバグ報告とアップデートを行います。ただ、残念ながら今回は誌面のみでの掲載になりますので、ご了承ください。

## バグ報告

とりあえず、現在までに確認されている代表的なバグを挙げてみましょう。

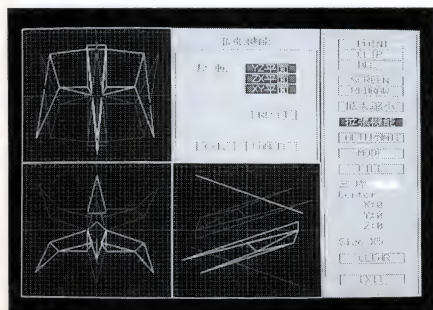
### ●拡大すると飛ぶ

拡大率が大きいなどで、透視図を大きくはみ出すような状態ではデータが腐ってしまうようです（または暴走する）。

これはどうもSLASH側の問題のようで、モデラで使用しているSLASHのバージョンが古いせいだと思われます。SLASHを新しいものに差し替えれば解消されると思われませんが、データ構造が若干違い、かなり大きな変更になってしまいますので、今回は対応しませんでした。拡大縮小などでは気をつけてください。

### ●拡張機能→対称

データの一部分が腐ってしまうバグがありました。今回のアップデートで解消されるはずですが。



対称機能を使うと

### ●ALTERNATE→合成

表ワークと裏ワークに同じ形状があった場合に、正しく合成できなかったようです。これも今回のアップデートで解消されるはずですが。

### ●FILE→PART

合成と同様、表ワークとファイルの内容に同じ形状があった場合に、正しく合成できなかったようです。合成と同じモジュールを呼んでいるので合成のバグが解消されていれば、こちらもちゃんと直っているはずですが。

### ●SCREEN

カレントポリゴンを適当に移動させると、表示がおかしくなったり、悪いときにはデータを破壊してしまっていたようです。今回のアップデートで怪しいと思われる部分は直したつもりです。また、オブジェクトがない状態ではSCREENモードに入れないようになりました。

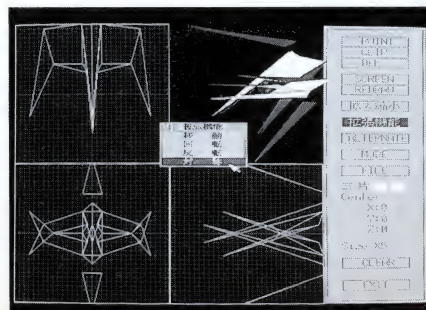
\* \* \*

### ●付加機能

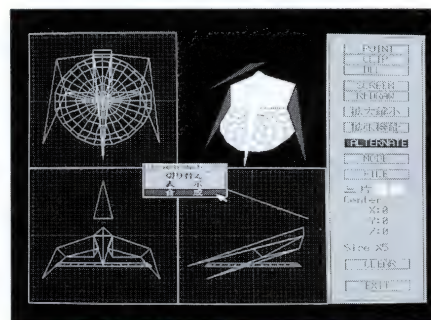
今回のアップデートにともない、比較的簡単にできる新しい機能もつけ加えておきました。

#### ・SCREENモード

‘-’キーでカレントポリゴンを削除します。確認はしてきませんので、気をつけてください。すべてのポリゴンを削除すると、その時点でSCREENモードを抜けます。



このようになる



裏画面との合成

## アップデートの方法

今月号には、先月号の付録ディスクに収録されていたmodel.xからの差分しか掲載されていません。したがって、SLASHモデラの最新版を作成するには、先月号の付録ディスク（を展開したもの）が必要です。まず、先月号の付録ディスクから解凍ディスク#6を作成するなどして、model.xを用意してください。また、model.xが大きいからといってlzxなどをかけてあるものは使用できません。

まず、今月号に掲載されているダンプリストをmac.xなどで入力し、model.lzhというファイル名でセーブしてください。セーブサイズは5522バイトです。セーブができたなら、

lha e model

として解凍してください。model.bfdというファイルができたはずですが。

そこで、今作成したmodel.bfdと先月号のmodel.xを同じディレクトリに置いて、

bup model

と入力してください。新しくできたmodel.xのタイムスタンプが93-9-22 22:29:24、ファイルサイズが110494バイトになっていたら成功です。新しくできたmodel.xをパスの通ったディレクトリに置くなどして使用してください。



ここで使用したmac.x,lha.x,bup.xは、先月号の付録ディスクから作成される解凍ディスク#1に収録されています。

## いいわけ

先月号で「次の機会にはちゃんと動くようにします」なんて大風呂敷敷広げたの

はいいのですが、まさか1カ月後に「次の機会」がくるとは思ってもみなかったの、はつきりいって今回は「とりあえず」状態になってしまいました。ごめんなさい（なんか、謝ってばかりだな）。

結局、先月号の原稿を出してからしばらくモデラは放ったらかしになってましたから（進歩のない奴）。まあ、今回のアップデ

ートでバグ出現度の高かったものはすべて修正できたはずですので、なんとかかとも使えるようになったんじゃないでしょうか。

では、次の正式版発表の機会にはちゃんと動くようにします（当たり前だという声が聞こえてきそうだが）。それではまた。

### MODEL.LZH

```
000000 22 CF 2D 6C 68 35 2D 6D : C1
000008 15 00 00 D5 B4 00 00 40 : DE
000010 5F 37 1B 20 01 09 6D 6F : B7
000018 64 65 6C 2E 62 66 64 E7 : 76
000020 AC 48 00 00 0C 6F 7A BB : A4
000028 1A 48 DB 7F EF FF CE D5 : 4D
000030 C7 1C 6C 57 04 22 1C 28 : 10
000038 C8 7E 24 9A 38 43 17 63 : F9
000040 53 86 24 AB 21 A2 BC 49 : 70
000048 C9 37 53 D8 BB 93 1B A5 : 39
000050 75 E9 04 F9 3F 02 7C B6 : CE
000058 A1 48 45 09 63 D4 D2 B0 : F0
000060 56 D7 29 C5 88 71 D4 31 : 19
000068 8B C0 BC 54 95 D4 5B 63 : 82
000070 91 5B 68 27 54 20 E8 5A : 31
000078 1D 5D 72 0E ED DB 2B 20 : 0D
```

SUM: 10 D2 9E D2 92 C2 E0 80 02B3

```
000080 C9 59 49 6C 44 18 CA E5 : E2
000088 75 6E 66 66 66 6E E6 E6 : 4F
000090 FF 99 99 B9 B9 99 B9 BF : B4
000098 F7 FD 8B 3B 16 CE 7A 6C : 84
0000A0 D7 D1 0A A8 84 3A 86 31 : CF
0000A8 D2 42 17 C1 0B E0 74 2F : 7A
0000B0 81 14 A1 69 4B E0 74 0B : 79
0000B8 EA A1 7C 14 A1 7C 77 C1 : 70
0000C0 7C 37 C7 7E 15 DF 80 AB : 17
0000C8 BB DE D6 48 DB 6D EC 71 : 5C
0000D0 02 0E F3 75 B7 EE B7 3A : 0E
0000D8 CF 7C 05 FF 39 8A 00 FE : 10
0000E0 6F 04 01 FC EF 0C 42 F6 : A3
0000E8 97 FF AB FA 5E D3 B0 FD : 19
0000F0 0F 8D 83 3B B7 FD E0 03 : F1
0000F8 F4 7E 66 93 72 7E 69 A6 : 6A
```

SUM: 59 D2 3B AA 4A 81 56 12 C194

```
000100 23 A4 31 2F C0 10 93 F2 : 7C
000108 3F 08 09 F9 89 0F C8 7B : 24
000110 9F EC 7E 10 12 F8 3F D0 : 32
000118 08 E5 01 2D C0 00 A6 8D : 0E
000120 3D 2A 3F 46 52 B6 80 97 : 3B
000128 14 86 F8 02 AB F9 1D 42 : 97
000130 1B D0 0A 90 02 AF 8D 3F : 02
000138 C0 4B 15 0E 14 40 53 72 : 47
000140 87 58 CF 20 8F 52 97 64 : AA
000148 9E 12 5A C1 27 FE 4C 38 : 74
000150 02 5A BA 34 80 44 1B 02 : 25
000158 22 EA 38 EA 6F C0 91 24 : 12
000160 4B 7E 60 29 09 69 6C 12 : 42
000168 E7 69 12 C4 76 A0 48 51 : D5
000170 43 D3 AF D6 99 D7 E9 35 : 29
000178 06 EA 12 F8 33 AF 47 4F : 72
```

SUM: F9 9A 57 05 1E C8 30 FD D412

```
000180 D8 24 35 81 32 26 90 D8 : 72
000188 97 0E 96 F1 3C 05 70 80 : 5D
000190 32 1A 25 33 00 24 81 EA : 33
000198 84 01 23 68 04 C7 C9 75 : 19
0001A0 09 0D 33 DC A9 43 5B 98 : 04
0001A8 4B BB EB 69 EA 94 25 D4 : D1
0001B0 CD 5A 85 FD 12 70 97 60 : 22
0001B8 91 3B 32 A1 F1 01 80 A5 : B6
0001C0 DB 34 24 55 00 84 2A 49 : 7F
0001C8 72 A1 07 44 93 2B CA 2D : 13
0001D0 BD 4F 6C AE F5 53 D2 A9 : E9
0001D8 4D AE 88 75 07 E5 0D 95 : 7E
0001E0 51 E0 C0 75 AA 8C 2D 76 : 3F
0001E8 28 A1 50 10 39 A9 20 26 : 51
0001F0 92 F1 24 B6 44 23 D6 9F : 39
0001F8 8A 90 EA 7F 8A 62 B0 12 : 2B
```

SUM: C3 76 1F 66 48 FF 87 29 EA28

```
000200 41 F0 02 5A A0 60 99 FC : 22
000208 0A 6A 0C AE 99 51 61 12 : 8B
000210 B6 9A 41 28 46 A1 D5 FC : 11
000218 B1 0B 12 8B 64 A3 20 D5 : 55
000220 F6 FA FC 67 57 F7 FC FE : 9B
000228 32 91 D1 F6 39 E5 BD 94 : F9
```

```
000230 5E EF 49 0B 92 59 B0 52 : 8E
000238 66 F9 8A 30 43 EC FE 38 : 7E
000240 F6 32 26 D0 35 03 13 00 : 69
000248 9B 38 84 AB AE 51 6A B2 : 58
000250 6C 6F 02 6C 79 4D 86 89 : 1E
000258 B5 C9 36 C4 AA 36 22 A6 : 20
000260 E9 B2 95 26 E7 01 26 E6 : 4A
000268 18 2A 4A AE 90 98 2A C4 : 50
000270 BD 67 95 2C A7 E2 5B 34 : FD
000278 4A 78 52 69 F7 F5 AB 51 : 65
```

SUM: 58 CF A9 67 9E FD D1 0B 7451

```
000280 8F 2E 68 FF FE C2 26 03 : 0D
000288 5A 60 AF 0C D4 75 14 3B : 0D
000290 39 25 14 82 49 37 64 A0 : 78
000298 79 CD D4 4B 51 FA 49 40 : 39
0002A0 21 B6 DB 75 A0 EB 65 EB : 02
0002A8 68 D3 4A 93 28 25 FF 57 : BB
0002B0 F6 AD 08 88 38 AB 62 54 : CC
0002B8 CE 4F D5 9F 98 7C 63 D0 : D8
0002C0 B3 82 1A 15 8F ED 03 47 : 2A
0002C8 A8 13 BC 0D 3D 8A FA 24 : 69
0002D0 7D 03 E3 BC 0E B6 43 C3 : AE
0002D8 90 D0 3A 33 FC 06 49 B2 : D7
0002E0 E2 C2 C2 F2 1E 74 B1 47 : E2
0002E8 E2 E1 26 83 33 60 78 : 0B
0002F0 8B B5 13 E2 4A 7C 09 7B : 7F
0002F8 59 AA F4 53 EB 06 93 EB : BC
```

SUM: F8 6F E3 C2 92 FC 46 8C 30EF

```
000300 08 4F 99 0A 62 9B 87 C1 : 3F
000308 4C 74 43 20 AA 3B 0F C4 : DB
000310 00 D6 33 B3 83 5D 29 B4 : 79
000318 9B 18 5E E2 40 A5 B9 98 : 29
000320 CE CA 0D 72 DA 97 91 C2 : DB
000328 7D 74 A5 7C 17 5B CA D1 : 19
000330 9A CE 14 3D B3 4B 6F 3D : 63
000338 F1 09 C6 18 DA 62 CA 41 : 1F
000340 28 8A 2B 4E 29 85 50 66 : 8F
000348 CD D3 D0 8F 4E 3E BF 06 : 50
000350 D2 E4 88 A9 AE 75 A2 B0 : 5C
000358 AE C3 63 0B 52 4A 9B 1A : 28
000360 7F 48 37 31 F6 CA BD 80 : 2C
000368 90 50 68 CA 23 3A 16 9B : 20
000370 4E BE D1 9E 96 BF 29 4B : 44
000378 AB 34 64 D9 9D 0B 1C 1B : FB
```

SUM: 3A 54 B3 05 10 C7 6A 99 48F6

```
000380 0C FC 2A A5 BE D7 07 59 : CC
000388 38 AF 46 C7 04 D2 7A B8 : FC
000390 A3 50 15 0A 6E E7 61 A6 : 6E
000398 6E B5 41 0F DB C2 A8 CB : 83
0003A0 ED B1 83 D4 8D E7 98 E9 : EA
0003A8 2C 1B 59 A3 D2 B2 E9 F9 : A9
0003B0 71 6B 2D 71 88 71 79 3D : 29
0003B8 3E 31 9D 3C 0A FB C4 23 : 34
0003C0 CE 48 48 03 40 AF 79 8F : 58
0003C8 3E 0E 84 F6 5D 6B 00 E2 : 60
0003D0 65 23 23 09 E2 21 44 AA : A5
0003D8 78 BD 73 BC 9A E9 2F DC : F2
0003E0 B1 81 D8 C0 44 CC 61 41 : 7C
0003E8 D7 D5 9A 31 D3 4A BE C5 : 17
0003F0 2F 5E FD 94 0B 67 8A 74 : 8E
0003F8 4B 77 3D D7 72 15 DE 02 : 3D
```

SUM: 08 79 7A B3 A9 0D BB 37 A3F6

```
000400 0E 62 1A E5 12 0C D6 DF : 42
000408 CA 1E 66 CA 92 39 66 B6 : F3
000410 59 60 53 72 57 63 0C 54 : 98
000418 A4 16 96 0A 66 5A AA 22 : E6
000420 F4 D9 B1 BA 1A 32 00 68 : EC
000428 FA 3E 6B CC 96 DA 71 5C : AC
000430 03 20 96 88 C5 53 11 D3 : 3D
000438 22 EE CC BC 9B 9A F3 16 : D6
000440 62 81 35 14 CD 87 00 A6 : 35
000448 FB 40 20 D5 D6 8D 6D D7 : D7
000450 8C D6 80 D2 02 1A C1 BB : 4C
000458 11 5B 11 29 D8 C8 62 6E : 16
```

```
000460 EB D3 63 3A 98 F5 20 41 : 49
000468 A6 66 2D EF DA 19 4E B9 : 22
000470 BF 82 22 75 3A 73 46 4B : 16
000478 35 C4 5A D7 D8 8E 68 E4 : DC
```

SUM: 61 8C D9 48 81 00 13 87 1340

```
000480 CD 35 E8 E8 08 5D 69 01 : A1
000488 62 67 B9 60 EA 97 85 21 : 09
000490 D7 94 4A 35 D0 F1 93 09 : 47
000498 BC 20 BF 0C 94 46 50 E2 : B3
0004A0 42 8E F0 0D 5D 40 7A 9C : 80
0004A8 5E FF E6 0F 09 F0 3D 3A : C2
0004B0 C1 F1 76 66 2F 1A 4F DC : 02
0004B8 23 6C 25 5F 45 2A 40 0E : D0
0004C0 37 A8 95 30 BE C7 FF D0 : F8
0004C8 66 6F FA 2A 40 8A 83 B2 : F8
0004D0 E2 EC E0 AC 7E F5 1F 57 : 43
0004D8 50 BE 6D A4 46 8A 93 FF : 81
0004E0 34 81 91 47 44 57 68 1D : AD
0004E8 73 B0 69 DD 5C E7 E0 D8 : 64
0004F0 73 74 51 B7 E6 D0 66 94 : 9F
0004F8 DD 36 C6 DE F6 EB DD E6 : 5B
```

SUM: 0C D6 08 CD 6E 68 D6 14 3CA9

```
000500 5C 8A 20 CA 19 E1 A2 52 : BE
000508 59 F1 7A 11 56 F3 A2 BB : 7B
000510 24 43 CF EC CD 18 D2 22 : FB
000518 C5 5B 0D 1F 07 69 7E C3 : FD
000520 4D 06 AC 52 D0 25 37 A5 : 22
000528 4E DD 50 5D EE D5 EE 50 : D9
000530 E6 8D EC F6 6B 81 D2 D8 : EB
000538 6F 17 92 D1 BF 17 29 47 : 2F
000540 8E 69 C3 28 17 D6 9D 97 : 03
000548 A1 BC C7 1E 15 F0 87 9F : 6D
000550 0D 5B 36 DC 4B 73 82 C7 : 81
000558 6F 51 CA 4C 0E F9 BF 40 : D6
000560 F6 BA AC FF 19 21 97 0B : D7
000568 4B 68 E7 AA 0E DB E7 69 : 4F
000570 9A 1F A5 2B 0E A0 47 55 : D3
000578 01 BE DA 03 31 5D F3 83 : A0
```

SUM: 15 70 2C 9B E8 12 D1 8F F584

```
000580 D2 40 38 90 04 76 B3 1D : 24
000588 F7 97 0D A1 72 C0 7E BB : A7
000590 1F A5 C5 69 AA DF 25 2E : CE
000598 DF 04 1E 65 BB 99 59 F6 : 09
0005A0 E4 9C 61 5C 2E 3B 38 A2 : 80
0005A8 B0 68 58 78 22 31 74 99 : 48
0005B0 CD B7 67 D1 8B 9D 7B 16 : 75
0005B8 72 E2 B9 E4 D9 0F 0D 07 : ED
0005C0 16 A5 87 7E 4A E7 F3 34 : 18
0005C8 82 21 40 79 63 7F 8F 2B : F5
0005D0 A4 D7 DB 6A EF 0B 31 DF : CA
0005D8 41 48 78 92 D6 21 A4 3A : 68
0005E0 4B 81 0C 6D 34 B5 62 0C : 9C
0005E8 22 E8 84 5E 64 EB 8A DA : 9F
0005F0 DC 37 96 2F 1F 44 AD A0 : 58
0005F8 07 93 D2 A0 1E 53 62 FD : DC
```

SUM: 67 35 13 15 D6 8C 35 4F B59A

```
000600 7E FB 9A 1F F6 64 BC A2 : EA
000608 21 C1 6A 2E D6 AA 71 9A : 05
000610 D9 DB BB 2A B1 DB 99 66 : 24
000618 E3 A3 51 FD 70 49 48 FA : CF
000620 24 A2 8E 5C 12 9D 61 15 : D5
000628 6A 84 5A EF E6 BD 3A 4A : 5E
000630 9D 1E 21 6D 2C B6 3B 0C : 72
000638 98 F1 A1 7E 24 33 12 12 : 23
000640 D2 A1 4F A4 25 8E BC BF : 34
000648 E9 63 78 BE C3 75 BA 23 : 91
000650 80 71 02 DC 47 D3 AD B3 : 49
000658 92 99 EE 8A 18 E7 FB 73 : 10
000660 AE 35 72 B4 09 9A 0C 3A : F2
000668 B0 CD AA 74 C9 9D 83 DC : 60
000670 EB B7 4F C9 40 CD 0E 53 : 28
000678 01 D2 5A FD 09 08 5D 63 : FB
```

SUM: 35 A8 36 60 97 3E 08 ED AD71



```

000680 12 5E D6 5D 93 A3 A3 5D : D9
000688 59 81 55 E1 1D 97 A1 13 : 78
000690 A1 2B 52 9B 66 8F 65 76 : 89
000698 68 D7 30 B4 E8 90 E6 33 : B4
0006A0 CE 53 8C 75 B5 F6 F4 A2 : 63
0006A8 85 74 08 6B A4 D2 81 A6 : AF
0006B0 99 77 68 EC 04 01 AF 3A : 52
0006B8 67 09 C8 AB 5F 8C AB 9C : 12
0006C0 E1 2C 40 B9 42 4E BC 9A : EC
0006C8 8B 18 BC 42 42 BB 0A 61 : 09
0006D0 31 CA D5 AE 20 5E 6C CC : 34
0006D8 E8 D1 4A 43 65 90 7C D0 : 87
0006E0 37 B9 DC 85 B0 9F 0D DA : 87
0006E8 3A C3 AC 92 B5 65 3A D5 : 64
0006F0 F1 96 02 CD AD 76 04 88 : 05
0006F8 D2 40 C4 D1 F4 C2 F4 F6 : 47

```

SUM: 80 59 DA A5 6F E1 48 FB 8F27

```

000700 33 6D D6 77 8E FB CE 6C : B0
000708 2C 50 E0 31 6F F4 34 22 : 46
000710 6F 70 73 A9 95 C7 69 A2 : 62
000718 2C F3 2B 9B 33 CE C5 1E : C9
000720 6D 1A 07 1A D7 EC FC DD : 44
000728 2F 77 B4 E3 80 3D 97 E3 : 74
000730 8B 7A 00 F7 BC 3E C4 DC : 96
000738 5F 97 DA 7D 20 49 DC 03 : 95
000740 45 A9 D9 79 A0 32 4E 38 : 98
000748 03 91 7B D9 7D 10 5F CC : A0
000750 90 7F CD F7 B6 F9 80 BC : BE
000758 F1 8C D8 7E A0 28 8B D9 : FF
000760 65 77 FF 3E 76 D6 76 5C : 37
000768 EC 20 61 49 C2 D7 9F BD : AB
000770 F0 D0 E2 76 B3 OF F9 07 : AF
000778 EC F7 68 70 B3 2F 51 C1 : DA

```

SUM: 76 65 8C 91 09 82 7A 67 0336

```

000780 F0 91 F9 E2 EC E7 64 72 : 05
000788 BD FF 9D 3F F2 76 7E 25 : A3
000790 1F 37 71 7C 7F CF 99 90 : BA
000798 09 1F 25 2C BD E7 ED E5 : EF
0007A0 F7 DB C9 3C 6E FB DD 82 : 9F
0007A8 44 9C B9 D8 7F 7B 7D 7B : 63
0007B0 B4 FE 51 19 79 91 20 08 : 4E
0007B8 7F 07 F6 01 E7 19 7C 0D : 06
0007C0 31 9C B0 65 07 F8 43 F8 : 1C
0007C8 46 6E 76 1C 74 00 71 21 : 4C
0007D0 DA F0 F1 E7 71 BB FC 8B : 55
0007D8 FD 3A 20 0E DB 7D 33 57 : 47
0007E0 A7 93 B6 DF 60 A2 26 E3 : DA
0007E8 F6 68 07 FB 89 33 0D D9 : 02
0007F0 7F 11 A0 F3 D2 1C BC E3 : B0
0007F8 FF 90 F9 EE E5 CE D5 CA : C8

```

SUM: AC 32 82 28 CE 22 05 82 431F

```

000800 FC FF 87 EE FE 84 42 9A : CE
000808 22 68 2F 37 74 FB 7F 0F : ED
000810 6D 3B 8D F0 F2 0D 67 A9 : 34
000818 50 9A CD BF CB EF E6 48 : 5E
000820 14 93 41 F5 0F FE A6 5D : ED
000828 3E 4C 84 39 C6 69 3E 1D : 75
000830 8A 1F 3D AC C3 F9 94 40 : 22
000838 1C 10 6D 78 1A A9 1F 37 : 2A
000840 7F 85 78 78 0A A2 F7 3E : D6
000848 82 25 57 E2 25 DA A1 9F : 1F
000850 95 C0 40 21 C5 DE 6B 8F : 53
000858 09 76 9F 10 FD 2E 8B 52 : 36
000860 8E 5F 0B 7F 84 62 9A 7C : 73
000868 19 5F 97 C9 D5 CA DB 7D : CF
000870 FD 21 77 B4 FA 46 83 00 : 0C
000878 FE 9F 5D 48 0B F6 BA 75 : 72

```

SUM: 14 A8 A3 F5 D5 74 E5 B7 6E28

```

000880 79 C4 44 61 49 CA E0 FC : D1
000888 F9 D8 C8 90 0C 5E EC 06 : 85
000890 62 CD D7 19 D7 53 47 0F : 9F
000898 2B 91 46 97 98 01 FA 7F : AB
0008A0 8C 00 7C 2F 7A B0 B5 26 : 3C
0008A8 58 35 33 00 6A 61 27 7A : 29
0008B0 0F BC 43 B7 C8 BF DB EF : 13
0008B8 50 FB 3F 0F 4F B5 53 C7 : B7
0008C0 03 66 8F 8E 44 9F 6D E8 : BE
0008C8 A5 A5 3F 0E 46 32 79 A1 : 29
0008D0 FA 94 8A EE 0E 01 FA 34 : 15
0008D8 D6 93 96 0E 5D 10 51 E1 : AC
0008E0 F6 68 11 4A 46 5A 4F 78 : 20
0008E8 00 80 CC 89 13 6F 66 19 : D6
0008F0 37 45 AA 02 04 DB 17 BC : DA
0008F8 01 98 52 54 16 FF 73 93 : 5A

```

SUM: E8 DD 21 57 F6 86 84 64 B8E3

```

000900 FB BC 9F 2B 81 85 23 0B : B5
000908 44 81 2E 3F 67 80 7F 4D : E5
000910 E0 24 87 86 D1 49 50 BD : 38

```

```

000918 BA 9D 27 5D F8 1B 59 8B : D2
000920 22 0A 3D 13 C7 6D 44 43 : 37
000928 47 15 30 09 24 3D 40 DF : 15
000930 E0 C9 F2 B7 F8 52 14 01 : B1
000938 B5 78 86 FE 27 EB AE B5 : 26
000940 FB 0E 18 64 1D A5 E3 7C : A6
000948 03 F9 EC 1C 9B ED 9C AD : D5
000950 C6 8C F8 20 7E 99 FE E6 : 65
000958 6E 3B 13 FC 7C DD 77 5C : E4
000960 79 83 F9 5A FD C8 2F 40 : 83
000968 66 BF 10 17 A7 E1 F9 2A : F7
000970 E1 DF 23 87 7C 7F BC 84 : A5
000978 F4 72 13 D1 F1 95 D1 C8 : 69

```

SUM: BD BF AE 83 7E 15 3A 99 CE2B

```

000980 47 47 20 FC 3F 08 F1 EC : CE
000988 F5 98 C6 E0 60 CA ED BB : 05
000990 4E 3F A0 82 11 64 7B 71 : 10
000998 A9 52 C2 C1 BA B3 DD 3A : 02
0009A0 0D 2D 20 B3 F2 B6 33 34 : 1C
0009A8 8C 8F B1 29 EE 37 C2 49 : 25
0009B0 8A 60 FC C4 E4 70 33 C8 : F9
0009B8 47 E7 CC B3 20 E5 17 7D : 46
0009C0 56 C7 C9 8A 2E CB 35 A6 : 44
0009C8 76 9A 69 40 33 13 13 95 : A7
0009D0 F4 FE 5E 2F 7B F1 7F 0E : 78
0009D8 61 1A 96 22 D7 29 34 13 : 7A
0009E0 8F 2E 8B 47 57 0D D8 38 : 03
0009E8 5B 4C B3 6E B6 9E F3 17 : 26
0009F0 F8 30 30 30 77 40 D2 A4 : B5
0009F8 8E 9B B3 D1 D3 77 A2 A6 : 3F

```

SUM: 2E 31 28 43 58 85 AF 09 7924

```

000A00 C1 CD F2 09 01 AA 0F CD : 10
000A08 DF A6 C2 04 AC 12 4B BF : B3
000A10 8D 2F E2 F1 D0 D3 A3 EB : C0
000A18 E0 76 12 FD E8 10 33 1F : AF
000A20 23 93 BA 00 3C 4F 7C 66 : DD
000A28 9D 1D 1E 1F 6F BB 07 C7 : EF
000A30 F1 B7 5A 79 3D BA 29 A5 : 40
000A38 22 64 99 B7 93 2F 8E ED : 13
000A40 FC 5E 0E E3 E0 E3 4E D9 : 35
000A48 CE C9 F3 B1 P6 5B 5D F7 : E0
000A50 C7 F1 B2 7E 72 5A F4 D7 : 2D
000A58 92 3A FD 52 BA F3 4F D7 : EE
000A60 C9 3B DD 78 BC 1D C9 98 : 93
000A68 DE EB 1A 76 3E C8 FD 9F : FB
000A70 29 3D 9C F4 BB 9F DA F9 : 23
000A78 49 05 FD C4 EC AE 0E F3 : AA

```

SUM: 1C 9D B3 54 23 4F 0E 9C E65A

```

000A80 97 C8 EF 91 63 6D EE 15 : B2
000A88 F2 41 5F 70 80 47 C3 47 : D3
000A90 E7 C3 EE 38 18 DB FC 6E : 2D
000A98 CB B8 45 61 BF 07 81 B5 : 25
000AA0 DF FC 8E 4E 93 24 14 7B : FD
000AA8 64 B8 63 BF 20 52 7F 3F : 6A
000AB0 FD C0 C0 3A B9 3D 9E ED : 1A
000AB8 BC D9 51 50 ED 80 D9 4B : C7
000AC0 24 6F 21 B3 05 0A C0 BA : F0
000AC8 B0 EA 3E 1D 04 72 C8 A5 : D8
000AD0 98 75 78 26 A7 BA 04 08 : 18
000AD8 B0 B3 E5 5D 85 C0 B2 10 : AC
000AE0 16 D1 F1 6D F9 9A F5 4A : 17
000AE8 57 64 90 60 AB 35 0A AD : 42
000AF0 42 B5 2C A6 80 A8 4E 32 : 71
000AF8 EA 22 AE 0E DC 84 AC 7D : 51

```

SUM: CE 5E 9A 05 48 BA 6B 8E B6B0

```

000B00 DE 66 B6 B9 01 11 6A 80 : AF
000B08 12 E7 D6 75 68 31 F3 47 : 17
000B10 5B 8F 85 74 4F BE 5C 13 : 65
000B18 A9 DA BC 43 56 3F 24 04 : 3F
000B20 39 D0 4D 68 DA 86 47 61 : C6
000B28 B2 D1 89 56 97 9C FA CC : 1B
000B30 03 05 8D 69 85 04 03 04 : 8E
000B38 E2 09 D3 A4 BF 5A 95 92 : A2
000B40 D8 92 89 25 0C 37 86 CE : AF
000B48 8A 4E 8B 15 2E 61 DB 08 : E7
000B50 B3 16 1A 35 48 95 A1 75 : 0B
000B58 E6 AD B3 08 01 5A 6F 05 : 17
000B60 DC 10 B9 F7 83 6C 38 D1 : 94
000B68 90 4E 6D D2 AD D3 04 C4 : 65
000B70 53 7B 32 DA E0 9F D8 00 : 2B
000B78 CD 8C 01 FB 44 54 45 71 : A3

```

SUM: 4B 6D 3D C5 5A 72 7D F7 9542

```

000B80 F6 F0 AF 71 FF 53 53 3A : E5
000B88 E1 A9 B8 A9 71 35 76 C6 : 73
000B90 30 45 19 9D 54 24 CA 11 : 7E
000B98 F0 EC D6 F4 2D 28 73 AD : 1B
000BA0 F3 D5 9A F6 74 30 B6 D5 : 87
000BA8 B1 86 3E 07 0E 2B 44 A3 : 9C

```

```

000BB0 2B BF 1A E5 C5 92 5D 72 : 0F
000BB8 7E 08 8A 6F 8D 9F 14 49 : 08
000BC0 CE 16 97 2C 4E A7 0E 0A : B4
000BC8 3C BE 57 38 12 74 38 8F : D6
000BD0 4E 46 57 88 A2 9C CB B7 : 53
000BD8 6E 63 BB 37 49 70 84 50 : 28
000BE0 88 E2 5A 12 6A E1 61 56 : D8
000BE8 FA E9 D7 99 19 87 8E CE : 4F
000BF0 A3 61 9C A6 D1 CC 39 8C : A8
000BF8 B8 59 2B C4 54 2B 97 11 : 27

```

SUM: DF EE CA 34 B8 E6 C5 F8 0C98

```

000C00 A6 86 55 6A B6 A0 49 59 : E3
000C08 5A E4 AA EF 40 66 EC 94 : FD
000C10 AC 48 10 6D 7B 94 14 E4 : 78
000C18 2A BA D7 D3 A0 54 D5 21 : 78
000C20 B8 62 ED CD 28 74 58 6B : 33
000C28 C9 24 EC 79 B9 6D 5C E1 : B5
000C30 7F 83 C4 24 04 97 18 DD : 7A
000C38 CC 76 DC F7 48 D4 73 D3 : 77
000C40 0F 56 FA 76 BB 44 9D 69 : DA
000C48 B6 B9 FD 9B 3E BB 78 E0 : 58
000C50 3D 74 80 3A 06 2E 4C 0D : F8
000C58 1A D4 30 E4 1F 2C 81 9C : 6A
000C60 4F 94 B5 9F F6 B8 18 6D : 6A
000C68 35 D4 5A 35 D7 81 95 D6 : 5B
000C70 AF 81 4B 91 7F 01 42 8D : 58
000C78 29 DA 15 F6 82 A9 77 1F : CF

```

SUM: 1A 05 75 84 2A 76 A5 CF 6966

```

000C80 09 0A AF 9D 64 D5 4F 6A : 51
000C88 19 16 6A E0 B3 80 DB 23 : AA
000C90 28 56 91 4A E4 5F 57 28 : 1B
000C98 73 55 9C BA DC A7 F2 A2 : 35
000CA0 AB 7D 03 32 44 5D F1 D0 : BB
000CA8 BA 09 78 6D 4E 89 F9 35 : AD
000CB0 27 7F 1D 4C CA D1 13 AC : 6A
000CB8 1A 1A 34 9E 6F 42 BF 07 : 7D
000CC0 2A D4 96 6E D6 1D 34 61 : 8A
000CC8 19 AF 23 89 41 5A 5C B1 : 1C
000CD0 D7 5F D2 CC 8C EA 82 91 : 57
000CD8 9C 8E 8D 6A 9A E3 50 AE : 78
000CE0 DC AE 6D 64 D7 B4 C5 F1 : 9C
000CE8 17 98 C7 84 D0 41 66 BD : 2E
000CF0 BA F1 29 80 D0 D3 A8 0D : AC
000CF8 0B 5D B3 F3 10 F4 F8 B0 : BA

```

SUM: C9 CE 3A 93 66 4E 5C CB B7C3

```

000D00 D0 FE F0 6E 45 4B 03 B1 : 70
000D08 01 B7 41 3B 4E 00 0B 8B : 18
000D10 76 97 71 2C B1 C4 06 84 : A9
000D18 54 1E E9 72 8C 27 94 BC : D0
000D20 FE 0E 3E DC 4E E6 30 E5 : 6F
000D28 AE 38 7A 0E FE 2B 2D F5 : 4E
000D30 2B 58 B5 C6 79 99 F3 4B : AD
000D38 08 BA 81 2F C6 D4 CC F2 : CA
000D40 2B 43 5A 6B 0C 9C E6 66 : 0F
000D48 F4 09 55 9B 3D 01 D7 6A : 6C
000D50 52 78 37 1F CC 4B 0F 7E : C4
000D58 B2 95 BD 17 E3 8D DA 79 : DE
000D60 E2 D5 8B 84 CE A2 EE F8 : 1C
000D68 31 36 13 EE 9A C6 DF 5B : FC
000D70 5A CA 7D 9A 8B 13 3A 8D : A0
000D78 21 CA 8A 67 8B DB 90 1B : 1A

```

SUM: 1F BA C1 D5 F8 7F E9 55 155F

```

000D80 20 E2 B2 CC 50 73 5A AA : 47
000D88 31 0D 50 B1 0B 40 8C 6E : 84
000D90 DA 5F 91 44 DE 57 43 70 : F5
000D98 D3 2B 1C 4A D6 41 10 30 : BD
000DA0 2D 1B F7 3D 6D 42 FE 30 : 59
000DA8 9A 2F 52 B8 89 F9 3C C8 : 53
000DB0 EE 26 41 45 04 34 07 F2 : CB
000DB8 4C 9A 8D 70 4E 75 8C 3D : 6F
000DC0 0E 19 AA 4E 9B 7C 8C 72 : 70
000DC8 79 B3 17 BB DD FF F7 6A : 3B
000DD0 1F B7 DA 8D 0B 72 AD 59 : C0
000DD8 DC 89 AE BD 06 AE AD 1C : 4D
000DE0 A8 CA CB 83 6B DB 6A 5F : 84
000DE8 86 82 7A 99 EE D8 A9 5A : E1
000DF0 5C 54 D3 22 CE 0B 98 60 : 76
000DF8 07 16 D5 6E 32 3B 0B EC : C4

```

SUM: 0C 05 FC B5 39 C3 D5 35 2522

```

000E00 59 76 A0 4F B5 6A A6 47 : CA
000E08 A8 61 D7 F1 B3 86 2A 3A : 6E
000E10 D5 A1 7D 8A CD 75 34 2B : 1E
000E18 80 86 BF 07 1D 59 B3 EE : E3
000E20 DE 14 A4 14 E1 F3 AE E3 : 0F
000E28 9D 77 60 22 82 06 8D 36 : E1
000E30 E5 74 EF 17 FA 66 37 89 : 7F
000E38 FE 37 B1 6D 58 70 18 45 : 78
000E40 CA FF 2B ED 1C F0 8B 2A : 9C

```

▶「SLASH」には、ずいぶんびびっときました。さっそくソースを解析します。

原稿 誠(18)神奈川県



000E48 ED 3E 1F B9 7D FD 70 4A : 37  
000E50 6F 3D 47 7A 21 20 8A 4F : 81  
000E58 A6 DD 53 9E 98 26 22 8F : E3  
000E60 53 73 EB 01 C1 57 87 3C : ED  
000E68 DA 3C 2C B4 48 65 9F 6E : A7  
000E70 9D 9C 36 2A 5E 4C 7D A2 : 62  
000E78 BE AF 5B 5A 68 B3 A3 E9 : C9

SUM: 02 7C E3 82 28 7B 8E 02 81C9

000E80 6A 58 D8 CC B5 09 6A 39 : C7  
000E88 68 22 7B B1 30 2A B5 96 : 5B  
000E90 8B 04 F1 0C CF 98 55 A3 : EB  
000E98 FF 78 13 61 A9 8F 06 DE : 07  
000EA0 64 6E 61 E8 C0 B8 75 B6 : BE  
000EA8 7A B5 78 62 B2 89 B7 40 : 3B  
000EB0 8A 7B 53 33 37 11 93 9A : FA  
000EB8 5C 19 BD 7F 3C 8A 86 DD : DA  
000EC0 1D 56 84 8E E6 F6 29 D1 : 5B  
000EC8 F8 3D 58 A8 3F EE 2A 05 : 91  
000ED0 0D B0 A0 44 1A 8F 32 75 : F1  
000ED8 BF E7 12 DA 03 20 C1 AB : 21  
000EE0 5B B1 FF 0E 88 EB 8F 05 : 20  
000EE8 0C F1 73 1C 84 BC 9F F2 : 77  
000EF0 8A 6B E0 EB AC 2E 85 3A : 59  
000EF8 7E 41 71 C1 CA 51 14 69 : 89

SUM: 70 25 91 08 0E EF E6 47 C5DF

000F00 86 B9 BC 98 8A 92 C2 7B : EC  
000F08 7F 3E BD 49 9C D2 AE 1C : 6B  
000F10 10 A3 86 83 15 42 AC 62 : E1  
000F18 B8 8A F6 BF AA 2C BD C2 : 4C  
000F20 CB 8A 16 8F EA 15 3F BB : F3  
000F28 40 68 EA AB 9C EC 04 8D : 53  
000F30 D3 5A 55 19 18 E0 49 4F : 2B  
000F38 9E BA 64 05 9B BE D6 C0 : B0  
000F40 9A D5 5C DC AC B7 30 91 : CB  
000F48 7D 43 D0 96 B5 D4 AB 9C : F6  
000F50 43 3C 31 D7 40 FC 72 D7 : 0C  
000F58 79 D0 4A 07 6C E3 FD DC : C2  
000F60 0C 4F 54 54 FF 81 30 D1 : 84  
000F68 AA 53 72 3F C1 5B D9 95 : 38  
000F70 94 7A 02 B7 78 66 00 33 : D8  
000F78 AE 2B 9A 17 58 F3 20 BF : B4

SUM: 14 95 B7 29 BB 10 DE 4A 0AD8

000F80 11 1C 50 D1 18 38 8A 39 : 61  
000F88 5A 57 65 F0 B0 ED CA 7C : E9  
000F90 D4 A6 B7 3B E7 AB AE CA : 76  
000F98 8E 7F 37 0C E2 1C 36 DE : 62  
000FA0 E3 26 DC B7 CA 1C C9 C0 : 0B  
000FA8 0D 69 11 49 C3 8F 83 9A : 3F  
000FB0 50 6A 95 66 46 FC 62 86 : DF  
000FB8 F5 7E 4D 88 17 65 59 B4 : D1  
000FC0 3D 9A 45 11 35 4E 64 5A : 6E  
000FC8 E0 0D 18 4A E2 C1 47 42 : 7B  
000FD0 CF A1 9A 5A 6E DE BF 5B : CA  
000FD8 1A 61 88 BC 9A BC CB 7A : 54  
000FE0 9E 9A 65 4B 2A 8C 3C CB : 2E  
000FE8 2F C9 32 DA 16 2B 3F 75 : F9  
000FF0 6C B8 7F ED 97 05 24 8D : DD  
000FF8 1C D9 72 AD 54 13 B3 2C : 5A

SUM: 5D AC 79 26 BF 72 4D 5B 4662

001000 B4 13 04 CB 5E 60 98 A6 : 92  
001008 9F FE 8E CA E7 FB 17 26 : 0E  
001010 E7 98 39 C3 FD 50 91 8B : E4  
001018 AA C4 4C 51 40 49 10 34 : D8  
001020 F8 AF 51 35 9F 0D 60 95 : CE  
001028 BE 74 27 03 61 7A EC 4A : 6D  
001030 88 61 E2 38 01 81 A3 F9 : 21  
001038 D3 85 EE 76 06 E3 E2 F6 : 7D  
001040 5C AB EE 58 B5 E6 8D 89 : FE  
001048 F7 FA 6C 20 61 5F 6A BD : 64  
001050 96 3E FF 0A F3 69 C7 00 : 00  
001058 76 B3 0E D1 62 F2 32 E7 : 75  
001060 6C 27 75 D7 82 9F FA C9 : C3  
001068 71 94 8A 8D 3F 9F F7 B : 68  
001070 81 F1 BE 89 FF C0 E5 A9 : 06  
001078 B0 3F A3 E5 C5 05 D4 E5 : FA

SUM: 62 F7 26 AE 79 76 C3 58 1717

001080 CD AB 5E 2F E4 BF AD A0 : 5B  
001088 7E B6 5D 7D 72 1F AB 07 : EE  
001090 F6 A8 AA 9E 0A BC 01 9E : 4B  
001098 5B 1D 85 ED 58 CD BE 80 : 4D  
0010A0 2F A4 03 AC F7 F9 94 B9 : BF  
0010A8 50 E8 0D A6 5B 15 D0 98 : C3  
0010B0 CC C8 06 39 DC 99 40 20 : A8  
0010B8 32 8E 58 BC 2F 46 B1 77 : 71  
0010C0 E5 A9 E1 87 AD C9 05 1D : 8E  
0010C8 80 B7 A0 0F 7B E0 1D E2 : 40  
0010D0 51 F2 35 BA CA 3E 46 2F : AF  
0010D8 8C 20 E5 CE CB DF 1F FF : 27

0010E0 E3 A1 8E 29 58 F8 BA 8F : D4  
0010E8 1C FF 1E 1E 1E B2 8E 5F : 14  
0010F0 93 93 94 A1 02 82 7F 25 : BB  
0010F8 9C DB DB B6 46 C9 A7 E1 : 9F

SUM: 89 88 0E 3A 90 0F 96 D4 B0E9

001100 43 67 98 8E 90 5D DE DD : 78  
001108 07 7A 2F 21 C8 89 5D 71 : F0  
001110 A8 89 C4 4B 22 AB C9 FF : D5  
001118 3E B1 26 9E BF 27 EC C0 : 45  
001120 1D DE B0 FF FE AF 51 C8 : 70  
001128 43 C4 5A F3 FE 7D AE A3 : 20  
001130 D0 3D 12 B4 68 74 99 68 : B0  
001138 13 74 3C 75 3A 33 F2 34 : CB  
001140 FA 8F 40 01 F8 3D AA 1E : C7  
001148 3D B3 64 FA 0B 75 65 2F : 62  
001150 15 AD 62 A7 22 22 CD 2D : 09  
001158 82 37 D6 E7 38 8C 17 11 : 62  
001160 65 65 10 C9 C5 7B 0A AD : 9A  
001168 36 85 D0 A5 9B 21 A3 ED : 7C  
001170 30 D4 5B 52 25 8C 77 AB : 84  
001178 E0 4A 5E A4 3D 9D 0B 62 : 73

SUM: EC 9C 7E A0 F6 B0 9C 46 D79B

001180 7C ED 9C B8 E1 FF CF 88 : F4  
001188 40 45 2A 26 8F 08 D1 E9 : EA  
001190 45 02 F2 AE FC EA 0F 56 : 32  
001198 19 82 99 B5 79 2B D6 45 : A8  
0011A0 01 14 B3 7D 7B 77 C3 48 : 42  
0011A8 73 46 AF E2 2C 38 CC 40 : BA  
0011B0 C8 28 17 33 FA 7A 3C A8 : 92  
0011B8 AE C1 FA 91 23 DC E1 44 : 1E  
0011C0 E9 20 50 31 94 FE 54 51 : C1  
0011C8 97 55 B0 2E DF 43 BC 6D : 15  
0011D0 AC D1 5F FE E8 82 6F 48 : FB  
0011D8 29 4E 94 11 9C C9 04 9A : 1F  
0011E0 6A F4 F7 93 BD 1B 6A B7 : FA  
0011E8 4F F0 8B AA 34 72 31 4D : 98  
0011F0 F5 E4 A7 CD 20 AB 55 16 : 83  
0011F8 71 DA CF 48 7A 34 09 3A : 53

SUM: 3C 2F AF 24 44 19 AD 74 5C93

001200 ED 17 16 D8 37 5D FD 2A : AD  
001208 74 F1 2E 43 15 5E DA E4 : 07  
001210 D6 EE 44 50 CC 59 B5 45 : 77  
001218 86 8A 96 GE 89 C0 EA 9A : E1  
001220 74 8E 6D EB A1 58 4E 13 : B4  
001228 75 29 8B F2 47 D2 B2 C9 : AF  
001230 00 D0 2C 37 4B EE F0 63 : BF  
001238 EF 80 0D A3 90 38 AD 03 : 97  
001240 DB ED 4D 67 56 75 B8 39 : 38  
001248 90 DD 71 22 39 F1 C2 DB : C7  
001250 D0 E0 48 80 D3 2D F2 06 : 70  
001258 58 A9 1A EE 39 A3 9D 9F : 21  
001260 35 4D 75 30 45 2F 59 86 : 7A  
001268 71 42 DA 68 D4 EE A3 AD : 07  
001270 42 D6 EC D1 9E 4C D5 CF : 62  
001278 38 B3 03 BD 01 EA B1 CA : 11

SUM: 48 F2 AD AD B7 AD 9E B3 6D0D

001280 96 4C 46 D4 6C 45 A8 54 : A9  
001288 6D 76 4C 33 75 4C 34 2B : 82  
001290 DB 6F 13 76 7A 6B D0 42 : CA  
001298 AD 0F 95 99 A0 58 7D 8B : EA  
0012A0 89 99 AE 81 68 DD C5 AC : 07  
0012A8 40 0D B3 55 25 B6 B3 4B : 2E  
0012B0 76 F0 05 D9 26 EC DE 8D : C1  
0012B8 23 51 20 B0 DA 57 AD C4 : E6  
0012C0 FC E6 22 F1 6D 39 38 51 : 24  
0012C8 71 FA 1A 18 E0 01 75 41 : 34  
0012D0 04 5A 37 46 8C DE 72 21 : D8  
0012D8 0A 03 42 E3 B1 6D 5A 74 : 1E  
0012E0 87 81 2B EA AA 5D D9 62 : 5F  
0012E8 B5 9D A6 D6 43 25 3B 10 : 81  
0012F0 80 2F 3D A5 F0 AF F6 9E : C4  
0012F8 F3 63 B8 A3 A9 37 0F F1 : 91

SUM: 17 14 3B AF 98 17 BE BC D517

001300 6A 54 77 9D 7E 60 0D 65 : 22  
001308 82 24 8F B1 A1 49 87 1E : 75  
001310 30 D9 33 6B 5E 6C 6A 32 : 0B  
001318 1A F8 01 AD A1 B1 73 05 : 8A  
001320 86 5D 86 86 74 E8 6D B3 : 6B  
001328 B4 93 11 1C 52 5D D3 DA : CF  
001330 AB 45 A6 17 E9 FA 1A D5 : 7F  
001338 92 9C FB A7 64 2B CF 53 : 81  
001340 75 6C 00 06 85 78 CE 1C : 8E  
001348 60 5D 89 AC CB 4E EB 3D : 33  
001350 0C 0C 08 D5 4A ED 88 FB : 63  
001358 9E B3 A6 43 E5 89 CE 67 : DD  
001360 D8 C1 79 71 50 7F 8A 9E : 7A  
001368 AB 49 75 55 0D 6A 1F 3B : 8F  
001370 10 B6 A2 0A 15 33 AD 51 : B8

001378 06 5E 6F 85 D3 6E C8 06 : 67  
SUM: 79 C0 68 E5 F3 F5 C7 5A 3EC0

001380 22 B4 39 C8 24 6C AB 95 : A7  
001388 76 3F 54 05 82 4A B9 86 : 19  
001390 E6 D7 9A 36 1B BA D5 D2 : 09  
001398 2D F8 F6 44 1B 05 81 33 : 33  
0013A0 EA 66 70 70 C4 F2 F6 2A : 06  
0013A8 3D 18 C9 2E CC 55 43 6D : 1D  
0013B0 33 96 C3 5A 9C E5 1F B6 : 3C  
0013B8 00 B4 7F 8F 05 27 40 D1 : FF  
0013C0 B6 63 51 69 DF DF 55 87 : 6D  
0013C8 EA 77 5A A1 D1 5E 1F FD : A7  
0013D0 AB B7 EC E7 00 1A 67 F7 : AD  
0013D8 7D 64 1C C7 47 32 AB 46 : 2E  
0013E0 99 B2 C4 A1 B3 90 9F DF : 71  
0013E8 2A 41 F6 E9 BA C1 02 FA : C1  
0013F0 AE 3D 5F 33 6A 5E B6 D1 : CA  
0013F8 82 44 90 49 07 66 0C 03 : 1B

SUM: C0 F3 F4 8C E2 64 3B AC D5DC

001400 F7 9A 45 2E 7F 52 BA 62 : F1  
001408 0D 31 FC 7E A8 FF 9B D3 : CD  
001410 FE EF 4F B5 3E 71 F2 B8 : 4A  
001418 5F 1F F7 7A 86 AF 4B 4A : B9  
001420 E4 BA F2 E6 79 E8 F6 72 : 3F  
001428 69 E5 FA 2A F6 5F C8 8D : 1C  
001430 C8 B4 41 1E 37 3A 87 FC : CF  
001438 6C FA 2A 1C 93 42 EF 52 : C2  
001440 E3 15 96 4B A9 CE A6 5B : 51  
001448 7E 31 EA 28 68 5A B4 9A : 89  
001450 16 E2 E4 3A 38 B7 2A 37 : 66  
001458 D0 36 49 6B 0D 8F 9D D1 : C4  
001460 A2 19 1D 35 2E 64 04 C4 : 67  
001468 38 0E BA 9E CA 1F 5A 4E : 29  
001470 EB 24 FF 90 80 18 DC 0E : 20  
001478 E7 6F DD 6E 7B 99 BB AD : 1D

SUM: D5 3E FC 0E 67 D6 DC 48 A6C2

001480 CC CD EA 47 97 C7 82 B0 : 5A  
001488 2D 5A C2 6F 7A B6 F2 07 : E1  
001490 6B 00 5B B2 42 B9 D0 B9 : E9  
001498 C8 FA DE A3 EC 6A D0 05 : 6E  
0014A0 B3 9B 38 95 CA 48 43 1D : 87  
0014A8 BF B0 5F 55 9B F7 39 EA : D2  
0014B0 44 B3 C0 34 59 73 74 35 : 60  
0014B8 88 6C F9 76 B7 50 D9 EA : 2D  
0014C0 4E 3F 0A E5 06 10 CA 16 : 72  
0014C8 A2 3E 14 AC 2C 5C 27 51 : A0  
0014D0 4A 97 08 53 5E 37 94 0A : 6F  
0014D8 E4 A8 BR 37 B9 39 ED 5B : B8  
0014E0 05 7D 13 9C C5 3B EB 6C : 88  
0014E8 A8 0A C0 49 3C 25 37 9F : F3  
0014F0 70 99 BA 54 FC BE 92 FD : 5A  
0014F8 1D 6C A2 20 90 DC 8D 07 : 4B

SUM: C2 D3 3F 13 85 78 7D 70 D271

001500 D0 55 66 C5 0B 6A 1F CE : B2  
001508 81 20 90 10 08 AE B1 14 : BC  
001510 76 26 8A 15 86 87 4C 61 : F5  
001518 B7 EB A0 D6 78 8D DD CE : C8  
001520 3F D6 88 0B FF 3D 7F BA : 17  
001528 DC DB 51 3F 60 05 80 23 : 4F  
001530 0F D9 24 B7 3B 7B 4B CB : 8F  
001538 A0 C1 7C F5 31 6E 3C 90 : 9D  
001540 D0 E9 2B CE 79 DB F5 3A : 35  
001548 77 4E 9F 9E 0F E6 AB FF : 9E  
001550 5B F0 5E 7D 64 01 0F D8 : 72  
001558 46 50 FF D2 CC DB 0F 6A : 87  
001560 5C 31 BA D7 0E FF 3F AD : E9  
001568 A9 BB 6D AA 44 29 E1 0E : D7  
001570 1A CA 68 45 41 12 20 28 : 26  
001578 68 75 0D DF 49 07 C8 AA : 8B

SUM: B7 6D 5C 16 A2 35 42 4B 7627

001580 A1 CE 63 69 A3 22 17 DD : F4  
001588 CF 73 EE 3E FB B5 3E E8 : 44  
001590 B8 00 00 00 00 00 00 00 : B8  
001598 00 00 00 00 00 00 00 00 : 00  
0015A0 00 00 00 00 00 00 00 00 : 00  
0015A8 00 00 00 00 00 00 00 00 : 00  
0015B0 00 00 00 00 00 00 00 00 : 00  
0015B8 00 00 00 00 00 00 00 00 : 00  
0015C0 00 00 00 00 00 00 00 00 : 00  
0015C8 00 00 00 00 00 00 00 00 : 00  
0015D0 00 00 00 00 00 00 00 00 : 00  
0015D8 00 00 00 00 00 00 00 00 : 00  
0015E0 00 00 00 00 00 00 00 00 : 00  
0015E8 00 00 00 00 00 00 00 00 : 00  
0015F0 00 00 00 00 00 00 00 00 : 00  
0015F8 00 00 00 00 00 00 00 00 : 00

SUM: 28 41 51 A7 9E D7 55 C5 3236



# モデリングの省力化のために 回転体生成プログラム

Tamura Kento 田村 健人

3Dシステムをうまく使うにはモデリングが重要になります。単純な形状生成はできるだけ自動化したいもの。ここでは定番ともいえる回転体を作ってみましょう。面の順番処理に注目してください。

私は頭を使うのが嫌いである。普段からできるだけものごとを考えなくても済むように行動している。プログラミングを趣味としてはいるが、コーディングするのに頭を使うような題材は意識的に避けている。

3D関係のプログラムといったら、1次変換のオンパレードであろう。さらにポリゴンともなると、陰面処理がどうか、法線ベクトルがどーのこーのとか、頭が痛い要素がたくさんなのである。もちろん私は3D関係のプログラムは敬遠してきた。

今回、SLASHのリリースにより、頭の痛い部分は最小限にとどめてポリゴンを扱えるようになった。3Dにしりごみする必要はない。不精者のこの私でさえ手を出さることができるようになったのだ。

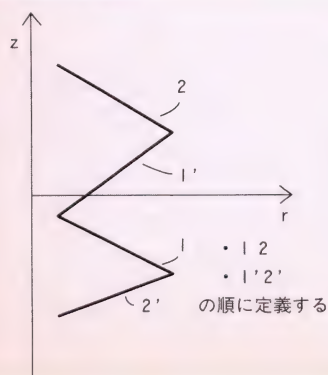
## 回転体を作る

表示のためのポリゴナイザはとりあえずある。とすれば、あとはモデリングをどのように省力化するかということが重要課題となってくる。

手作業でモデリングするためのモデラはすでに作成されているので、ここではコンピュータを使ったほうがよい例のひとつとして回転体モデラを制作してみよう。

回転体とはどのように作られるのかとい

図1 面の定義順序



うところから考えてみる。まず、断面図を描き適当に分周して、

$$x = r \cdot \cos \theta$$

$$y = r \cdot \sin \theta$$

として点を生成し、これらを適切に結んだ平面を作っていけば回転体ができるだろう。きわめて簡単である……はずだった。

実はSLASHは高速化のため、興味深い特徴を持っている。

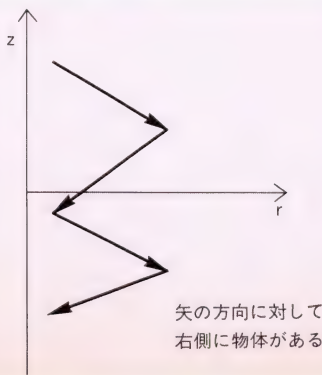
面は定義した順に描画されるのである。つまり、先に定義した面ほど奥に描画されるのである。図1を見ていただきたい。このように、前後関係によって定義順序を決めなければならない。

断面を構成する直線の傾きなどで、定義順序を決められるように思える。その程度ならば、さほど頭を使わなくてもプログラムできるだろう。私はそう思って回転体生成プログラムを作ることにしたのである。

では、具体的にどういったアルゴリズムで定義順序を求めるのかと考えてみると、傾きで分類する程度では不可能であることがわかってしまった。図1では傾きが負になる直線と正になる直線の間では定義順序が任意になる。だからといって、どんな形状でもそうなるとは限らないのだ。

ところで、SLASHでこういった定義順序の決まりがあることは開発中からわかり

図2 内と外を決める



きっていることなので、\_slashlib.aにはSortPoly()という関数が入っている。この関数は、面の位置や向きなどを調べて、正しく表示されるように定義順序を変える関数だ。ということは、ひととおりポリゴンの定義が終わってからこの関数を呼びばいいではないか。

はじめはSortPoly()を使ってやってみたのだが、使い方が悪いのかうまくいかなかったのである。よく話を聞くとまだ未完成とのこと。

結局、直線の方程式とかベクトルとか頭の痛い分野を使って、地道に前後関係を調べるしかなくなってしまった。ああ、面倒くさい。ここ以降は読むのがつらいだろうが、書くほうはもっとつらい。

## 有向線分の位置関係を調べる

SLASHには面の定義順序のほかにもうひとつ特徴がある。面の表裏があり、裏からはその面が見えないのである。よって、どっちが表なのかということも考慮してやらなければならない。

この回転体生成プログラムでは、断面を時計回りで指定することにする。断面を構成するのは有向線分となり（つまり、向きがあり、範囲が有限な直線）、その左側が表面になるのである。もし反時計回りで指定すると、回転体の中身に視点を置かないと見えなくなる。

いちばん手前に表示されるべき面を最後に定義すればいい。ある有向線分に対して、ある有向線分が手前にくるかどうかを判定するのが処理の要である。それさえできれば、いちばん手前にくる面（を生成する有向線分）を探し出すことは造作もない。

では、任意の2本の有向線分の前後関係を調べる方法を図3に示す。それぞれの有向線分の左側から視線がくるということを念頭に置いて見てもらいたい。



図3を見ると、逆向きの線分では無関係と  
いうことになっている。しかし、図4の断面  
から生成される形状においては、外側の壁  
面が、向こう側の内側の壁面より手前  
にののわかるだろうか。このように回転体  
であるのだから、片方の線分を回転させた  
あとの前後関係も調べなければならない。  
回転させたほうの右向線分は、右側からし  
か見えなくなる。

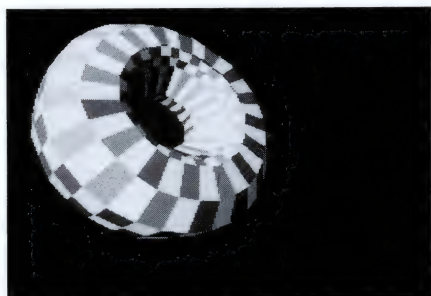
これらをプログラムにする。ひと苦労で  
ある。

## 使い方

掲載したリストすべてを打ち込んで、  
makeすればslround.xというファイルが  
できる。gcc+libcの環境でのみコンパイル  
確認した。

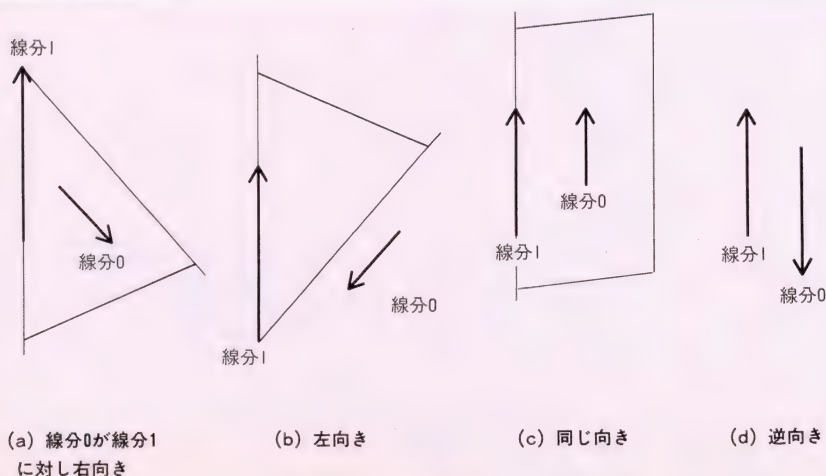
A>slround bar.plg

とすると、画面に横線が描かれる。この線  
が $z=0$ 平面である。画面の上半分が $z<0$ 、  
下半分が $z>0$ となる。画面の左端が半径=  
0である。マウスの左クリックで点を順番に  
指定し、左右クリックで終了する。点は回  
転体の断面を時計回りでたどるように指定

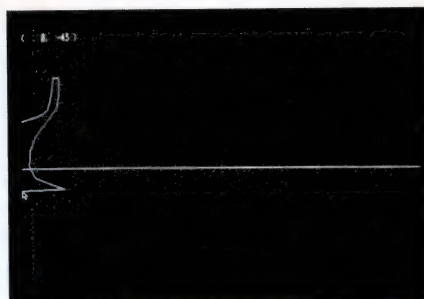


このように……

図3 有向線分を調べる



線分0が色のついた部分に  
かかっていると、線分1のほうが手前



左端を回転軸として点を指定

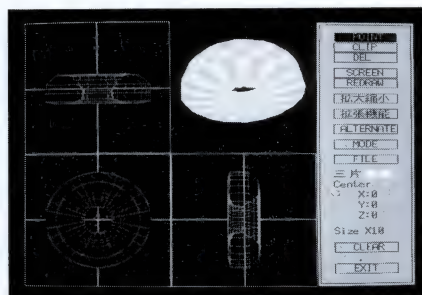
すること。右クリックで線分を1本キャン  
セルできる。このへんのユーザーインタフ  
ェイスは少し不親切だが、本題ではないの  
で勘弁していただきたい。終了したら、指  
定したファイル(この場合は bar.plg)が生  
成される。

できたplgファイルは先月号の付録ディ  
スク「秋祭りPRO-68K」に収録されていた  
model.x, testplg.batで見るができる。

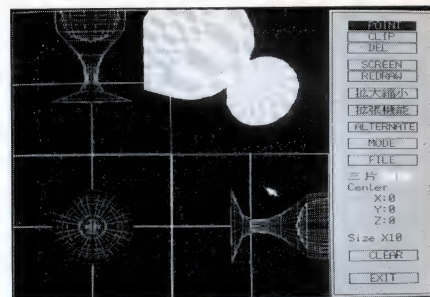
## 問題点など

このように苦労して作った回転体モデラ  
だが、いくつか欠点もある。

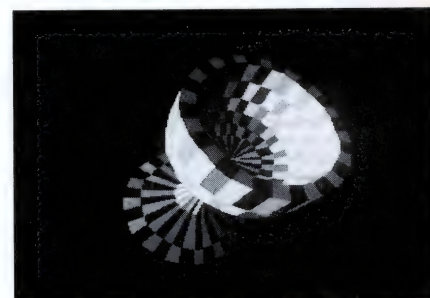
- ・半径が小さいと正常に描画されない



穴が開いても大丈夫



モデラを読み込んだところ



testplgで陰影を確認する

半径の値が1など、0以外の極端に小さい  
値が設定されると、正常に描画されない。  
 $r \cdot \cos \theta$ ,  $r \cdot \sin \theta$ の計算で切り捨てられる誤  
差が無視できなくなり、SLASHが面の向  
きを決定できなくなるのである。これは防  
ぎようがない。

- ・面の色を指定できない

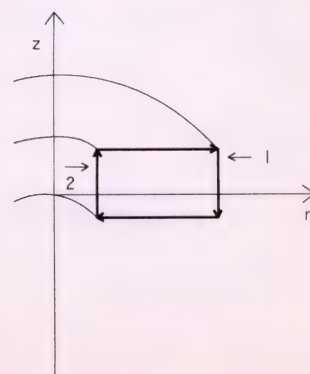
ユーザーインタフェースを考えるのが面  
倒だったため、面の色は15と14を交互に置  
いている。

- ・分解能が固定

これもユーザーインタフェースを考える  
のが面倒だった。いまのところ、32分周で  
固定である。実用上は、もう少し粗いほう  
がいいと思われる。

これらの問題は実際に使用する人によっ  
て拡張されるのが望ましい。基本部分はす  
べて作成されているので、適宜改造して活  
用してほしい。

図4 面1と面2の順番は?





# リスト1 round.c

```

1: /* Oh!X '93 11月号 slash特集 回転体作成プログラム 本体
2: by けん と */
3:
4: #include <stdio.h>
5: #include <stdlib.h>
6: #include <string.h>
7: #include <math.h>
8:
9: #include <_slashlib.h>
10:
11: #undef DEBUG
12:
13: /* 構造体定義 */
14:
15: typedef union {
16:     struct {
17:         short r;
18:         short z;
19:     };
20:     unsigned int i;
21: } POINT2;
22:
23: typedef struct {
24:     unsigned short kind;
25:     unsigned short npoint[4];
26:     unsigned short color;
27: } PLGPOLYGON;
28:
29: /* 大域変数 */
30:
31: unsigned short 分解能 = 32; /* 3~4095 */
32:
33: /* 前宣言 */
34:
35: int makeplg( POINT2* ptl, char* splg );
36: int writelpl( char* splg );
37: int 手前ですか? ( POINT2* ppt, unsigned short ps0, unsigned short pe0,
38:     unsigned short ps1, unsigned short pe1 );
39:
40: /* コード始まり */
41:
42: #ifdef HAVEMAIN
43: int main( int argc, char* argv[] ) {
44:     POINT2 ptl[100] = { 0 }; /* エントコードとして 0x80008000 を使う */
45:     /* 10,-60, 50,-60, 30,0, 50,60, 10,60 */
46:     /* 10,60, 30,60, 50,40, 30,20, 50,0, 30,-20, 50,-40, 30,-60, 10,-60,
47:     50,-80,0,8000,
48:     );
49:     return makeplg( ptl, "foo.plg" );
50: }
51: #endif
52:
53: int makeplg( POINT2* ppt, char* splg ) {
54:     unsigned short i,j,k;
55:     /* (半径線の数+1)x分解能 の頂点の点を生成する */
56:     /* 半径線の数+1 == それを構成する点の数 */
57:     SLPOINT* psplt;
58:     SLPOINT* psplt0;
59:     /* 半径線の数x分解能 の頂点の面を生成する */
60:     PLGPOLYGON* pplg;
61:     PLGPOLYGON* pplg0;
62:     /* 線を表す点の組み */
63:     unsigned short* pline; /* ppt 内のインデックスで表す */
64:     unsigned short* pline0;
65:     POINT2* ppt0 = ppt;
66:     short ipt = 0; /* 半径線の頂点 */
67:     /* を、載える */
68:     while ( ppt0->i != 0x80008000 ) {
69:         ppt0++;
70:         ipt++;
71:     }
72:     (void*)psplt = malloc( ipt*分解能*sizeof(SLPOINT) );
73:     (void*)pplg = malloc( (ipt+1)*分解能*sizeof(PLGPOLYGON) );
74:     (void*)pline = malloc( (ipt+1)*2*分解能*sizeof(unsigned short) );
75:     if ( !psplt || !pplg || !pline ) {
76:         fprintf( stderr, "メモリを確保できません ヒープを増やして起動して下さい\n" );
77:         free( psplt );
78:         free( pplg );
79:         free( pline );
80:         return 1;
81:     }
82:     /* 点を作る */
83:     ppt0 = ppt;
84:     psplt0 = psplt;
85:     for ( i=0; i<ipt; i++ ) {
86:         for ( j=0; j<分解能; j++ ) {
87:             psplt0->z = ppt0->p.z;
88:             psplt0->x = ppt0->p.r * cos( 2*M_PI*j/分解能 );
89:             psplt0->y = ppt0->p.r * sin( 2*M_PI*j/分解能 );
90:             psplt0++;
91:         }
92:         ppt0++;
93:     }
94:     /* 線を表す点の組み */
95:     /* 他どの面よりも手前にあるべき面を生成する線を */
96:     /* 一番後にスワップする */
97:     /* 一番後を探すのが 最大 O(n^2) */
98:     /* さらにループするので、結局 O(n^3) */
99:     for ( i=ipt-1; i>0; i-- ) {
100:         for ( j=0; j<i; j++ ) {
101:             unsigned short 手前 = 0;
102:             for ( k=0; k<i; k++ ) {
103:                 if ( j != k ) {
104:                     /* 2本目が1本目より手前だったら1を返す */
105:                     手前 += 手前ですか? ( ppt, pline[2*j], pline[2*j+1], pline[2*k],
106:                         pline[2*k+1] );
107:                 }
108:             }
109:             if ( 手前 ) {
110:                 /* 手前 */
111:                 fprintf( stderr, "%d\n", 手前 );
112:             }
113:         }
114:     }
115: }
116:
117: /* 面を作る */
118: pplg0 = pplg;
119: pline0 = pline;
120: for ( i=0; i<ipt-1; i++ ) {
121:     k = *pline0++;
122:     pline0++;
123:     for ( j=0; j<分解能; j++ ) {
124:         pplg0->kind = 1; /* 四角形 */
125:         pplg0->npoint[0] = k*分解能+j;
126:         pplg0->npoint[1] = (k+分解能-1)*分解能+j;
127:         pplg0->npoint[2] = k*分解能+(j+1);
128:         pplg0->npoint[3] = (k+分解能-1)*分解能+(j+1);
129:         /* 面の色はてきとでず */
130:         pplg0->color = (i+j)*2+1;
131:         if ( (psplt[pplg0->npoint[3]].x*psplt[pplg0->npoint[3]].y == 0) ) {
132:             pplg0->kind = 0; /* 三角形 */
133:             pplg0->npoint[3] = pplg0->color;
134:         }
135:         if ( (psplt[pplg0->npoint[0]].x*psplt[pplg0->npoint[0]].y == 0) ) {
136:             pplg0->npoint[1] = pplg0->npoint[2];
137:             pplg0->npoint[2] = pplg0->npoint[3];
138:             pplg0->kind = 0;
139:             pplg0->npoint[3] = pplg0->color;
140:         }
141:         pplg0++;
142:     }
143: }
144:
145: /* *.plg で出力 */
146: {
147:     FILE* pf;
148:     unsigned short w;
149:     pf = fopen( splg, "wb" );
150:     if ( !pf ) return 2;
151:     fprintf( "PLG MODELER v1.00 X05%0a%0a", 1, 24, pf );
152:     w = ipt*分解能;
153:     fwrite( &w, 1, sizeof(unsigned short), pf );
154:     fwrite( psplt, 1, ipt*分解能*sizeof(SLPOINT), pf );
155:     w = (ipt-1)*分解能;
156:     fwrite( &w, 1, sizeof(unsigned short), pf );
157:     for ( i=0; i<ipt-1; i++ ) {
158:         fwrite( &pplg[i], 1, sizeof(PLGPOLYGON)-(pplg[i].kind==0)*sizeof(short), pf );
159:     }
160:     fclose( pf );
161:     free( psplt );
162:     free( pplg );
163:     free( pline );
164:     return 0;
165: }
166:
167: /* 境界値付近は実はどうでもいい */
168: return t;
169:
170: double 正規化( double t ) {
171:     /* 値を -π <= t < π に収める */
172:     if ( t < -M_PI ) return t+2*M_PI;
173:     if ( t >= M_PI ) return t-2*M_PI;
174:     /* -π, 0, π のときは叫ばれないので、 */
175:     /* 境界値付近は実はどうでもいい */
176:     return t;
177: }
178:
179: int sgn( int i ) { /* libc には sgn() が無いように */
180:     if ( i ) {
181:         if ( i > 0 )
182:             return 1;
183:         else
184:             return -1;
185:     } else {
186:         return 0;
187:     }
188: }
189:
190: #define INLINE1( X, Y ) ((dy0*(X)-dx0*(Y)+dx0*ppt[ps1].p.z-dy0*ppt[ps1].p.r)
191: #define INLINE1_1( X, Y ) ((dy1*(X)-dx1*(Y)+dx1*ppt[ps1].p.z-dy1*ppt[ps1].p.r)
192: #define INLINE1E( X, Y ) ((dy0*(X)-dx0*(Y)+dx0*ppt[ps1].p.z-dy0*ppt[ps1].p.r)
193: #define INLINE1E_1( X, Y ) ((dy1*(X)-dx1*(Y)+dx1*ppt[ps1].p.z-dy1*ppt[ps1].p.r)
194: #define INLINE0IS( X, Y ) ((dy0*(X)-dx0*(Y)+dx0*ppt[ps1].p.z-dy0*ppt[ps1].p.r)
195: #define INLINE0IS_1( X, Y ) ((dy1*(X)-dx1*(Y)+dx1*ppt[ps1].p.z-dy1*ppt[ps1].p.r)
196: #define CV0( X ) ((dy0)*(X)/dy0):((X)/dx0)
197: #define CV1( X ) ((dy1)*(X)/dy1):((X)/dx1)
198:
199: int 手前ですか? ( POINT2* ppt, unsigned short ps0, unsigned short pe0,
200:     unsigned short ps1, unsigned short pe1 ) {
201:     short dx0, dy0, dx1, dy1;
202:     double θ0, θ1, dθ;
203:     #ifdef DEBUG
204:     fprintf( stderr, "(X3.3d,X3.3d)->(X3.3d,X3.3d)%s", ppt[ps0].p.r, ppt[ps0].p.z,
205:         ppt[ps0].p.r, ppt[ps0].p.z );
206:     fprintf( stderr, "(X3.3d,X3.3d)->(X3.3d,X3.3d)%s", ppt[ps1].p.r, ppt[ps1].p.z,
207:         ppt[ps1].p.r, ppt[ps1].p.z );
208:     #endif
209: }

```



```

246: dx0 = ppt[pe0].p.r-ppt[pe0].p.r;
247: dy0 = ppt[pe0].p.z-ppt[pe0].p.z;
248: dx1 = ppt[pe1].p.r-ppt[pe1].p.r;
249: dy1 = ppt[pe1].p.z-ppt[pe1].p.z;
250:
251:
252: 	θ0 = atan2( dy0, dx0 );
253: 	θ1 = atan2( dy1, dx1 );
254: 	dθ = 正規化( θ1-θ0 );
255:
256: 	/* 線分が交差している場合 */
257: 	if ( ps0==ps1 || ps0==pe1 || pe0==ps1 || pe0==pe1 ) goto NC;
258:
259: 	/* ちょっと逆方向の場合 */
260: 	if ( dy0*dx1==dy1*dx0 && sgn(dx0)!=sgn(dx1) ) goto NC;
261:
262: 	/* 同じ向きの場合 */
263: 	if ( dy0*dx1==dy1*dx0 ) {
264: 		if ( ( θ1 > 0 && CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z )) > 0 ) ||
265: 			( θ1 <= 0 && CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z )) < 0 ) ) {
266: 			return 1;
267: 		} else {
268: 			goto NC;
269: 		}
270: 	}
271: #ifdef	DEBUG
272: 	printf( stderr, "θ0: %f, θ1: %f\n", θ0, θ1 );
273: #endif
274: 	if ( dθ > 0 ) {
275: 		/* 線分0 は 線分1 より右側に入っている */
276: 		int inline1, inline0;
277: 		inline1 = CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z ));
278: 		inline1 = (θ1<0) ? -inline1 : inline1;
279: 		inline0 = CV0(INLINE0IS( ppt[pe0].p.r, ppt[pe0].p.z ));
280: 		inline0 = (θ0<0) ? -inline0 : inline0;
281: 		if ( ( inline1 > 0 ) && ( inline0 > 0 ) ) {
282: 			return 1;
283: 		} else {
284: 			int inline1, inline0;
285: 			inline1 = CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z ));
286: 			inline1 = (θ1<0) ? -inline1 : inline1;
287: 			inline0 = CV0(INLINE0IS( ppt[pe0].p.r, ppt[pe0].p.z ));
288: 			inline0 = (θ0<0) ? -inline0 : inline0;
289: 			inline0 = (θ0<0) ? -inline0 : inline0;
290: #ifdef	DEBUG
291: 			printf( stderr, "%d,%d\n", inline1, inline0 );
292: #endif
293: 			if ( ( inline1 > 0 ) && ( inline0 > 0 ) ) {
294: 				return 1;
295: 			}

```

```

296: 	}
297:
298: NC:
299: 	/* 線分1を回転させてもう一回 */
300: 	dx1 = -dx1;
301: 	θ1 = atan2( dy1, dx1 );
302: 	dθ = 正規化( θ1-θ0 );
303:
304: 	if ( dy0*dx1==dy1*dx0 && sgn(dx0)==sgn(dx1) ) goto NC2;
305:
306: 	if ( dy0*dx1==dy1*dx0 ) {
307: 		if ( ( θ1 > 0 && CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z )) < 0 ) ||
308: 			( θ1 <= 0 && CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z )) > 0 ) ) {
309: 			return 1;
310: 		} else {
311: 			goto NC2;
312: 		}
313: 	}
314:
315: #ifdef	DEBUG
316: 	printf( stderr, "θ0: %f, θ1: %f\n", θ0, θ1 );
317: #endif
318: 	if ( dθ > 0 ) {
319: 		int inline1, inline0;
320: 		inline1 = CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z ));
321: 		inline1 = (θ1<0) ? -inline1 : inline1;
322: 		inline0 = CV0(INLINE0IS( ppt[pe0].p.r, ppt[pe0].p.z ));
323: 		inline0 = (θ0<0) ? -inline0 : inline0;
324: 		if ( ( inline1 < 0 ) && ( inline0 > 0 ) ) {
325: 			return 1;
326: 		} else {
327: 			int inline1, inline0;
328: 			inline1 = CV1(INLINE1( ppt[pe0].p.r, ppt[pe0].p.z ));
329: 			inline1 = (θ1<0) ? -inline1 : inline1;
330: 			inline0 = CV0(INLINE0IS( ppt[pe0].p.r, ppt[pe0].p.z ));
331: 			inline0 = (θ0<0) ? -inline0 : inline0;
332: 			inline0 = (θ0<0) ? -inline0 : inline0;
333: #ifdef	DEBUG
334: 			printf( stderr, "%d,%d\n", inline1, inline0 );
335: #endif
336: 			if ( ( inline1 < 0 ) && ( inline0 > 0 ) ) {
337: 				return 1;
338: 			}
339: 		}
340: 	}
341: NC2:
342:
343: 	return 0;
344: }

```

## リスト2 ui.C

```

1: /* Oh!X '93 11月号 slash特集
2: 回転体生成プログラム ユーザインタフェース部
3: 	by けんと */
4:
5:
6: #include <ioclib.h>
7: #include <stdio.h>
8:
9:
10: typedef union {
11: 	struct {
12: 		short r;
13: 		short z;
14: 	} p;
15: 	unsigned int i;
16: } POINT2;
17:
18:
19: int makeplg( POINT2* ptl, char* splg );
20:
21:
22:
23: int main( int argc, char* argv[] ) {
24: 	int msdata, msdata1;
25: 	POINT2 ptl(100);
26: 	int ip = 0;
27: 	struct LINEPTR lr = { 0, 256, 767, 256, 15, 0xffff };
28: 	char buf[100];
29:
30: 	CRIMOD( 16 );
31: 	B_CUROFF();
32: 	G_CLR_ON();
33: 	MS_INIT();
34: 	MS_LIMIT( 0, 0, 767, 511 );
35: 	MS_CURON();
36: 	SKEY_MOD( 0, 0, 0 );
37: 	LINE( &lr );
38: 	msdata = MS_GETDT();
39: 	msdata1 = 0;
40: 	while ( -1 != (short)msdata ) {
41: 		int i = MS_CURDT();
42: 		B_LOCATE( 0, 0 );
43: 		sprintf( buf, "( %d, %d )", i>>16, 256-(short)i );
44: 		B_PRINT( buf );
45: 		if ( (msdata1&0x0000ffff)&&!(msdata&0x0000ffff) ) {

```

```

46: 		if ( !ip || (ptl[ip-1].p.r!=i>>16 || ptl[ip-1].p.z!=256-(short)i) ) {
47: 			ptl[ip].p.r = i>>16;
48: 			ptl[ip].p.z = 256-(short)i;
49: 			ip++;
50: 			if ( ip != 1 ) {
51: 				lr.x1 = ptl[ip-2].p.r;
52: 				lr.y1 = 256-ptl[ip-2].p.z;
53: 				lr.x2 = ptl[ip-1].p.r;
54: 				lr.y2 = 256-ptl[ip-1].p.z;
55: 				lr.color = 15;
56: 				LINE( &lr );
57: 			}
58: 		}
59: 		if ( (msdata1&0x000000ff)&&!(msdata&0x000000ff) ) {
60: 			if ( ip > 1 ) {
61: 				lr.x1 = ptl[ip-2].p.r;
62: 				lr.y1 = 256-ptl[ip-2].p.z;
63: 				lr.x2 = ptl[ip-1].p.r;
64: 				lr.y2 = 256-ptl[ip-1].p.z;
65: 				lr.color = 0;
66: 				LINE( &lr );
67: 				ip--;
68: 				MS_CURST( ptl[ip].p.r, 256-ptl[ip].p.z );
69: 			}
70: 		}
71: 		msdata1 = msdata;
72: 		msdata = MS_GETDT();
73: 	}
74: 	MS_CUROFF();
75: 	SKEY_MOD( -1, 0, 0 );
76: 	B_CURON();
77: 	CRIMOD( 16 );
78:
79: 	if ( ip ) {
80: 		ptl[ip].i = 0x80008000;
81: 		if ( argc == 1 ) {
82: 			makeplg( ptl, "foo.plg" );
83: 		} else {
84: 			makeplg( ptl, argv[1] );
85: 		}
86: 	}
87:
88: 	return 0;
89: }

```

## リスト3 Makefile

```

1: #!/bin/make
2:
3: # かいてんたい作成プログラムの Makefile
4: 	by けんと
5:
6: # 著者の環境
7: # HUMAN.SYS v3.01
8: # zsh v2.3.1 X6.03p6 Paul Falstad / 小野秀貴
9: # gcc v1.13 based on 1.42 F.S.F. / 真里子
10: # has v2.53 YUNK
11: # hlk v2.28 SALT
12: # GNU make v3.62 X6.10 F.S.F. / homy
13: # libc 1.1.28 Project LIBC
14: # (敬称略)
15:
16: OBJJS = roundm.o
17: OBJJS2 = ui.o round.b

```

```

18:
19: CC_OPTION = -O -Wall -fomit-frame-pointer -fno-bss
20:
21: all: slround.x
22: #all: slround.x round.x
23:
24: #round.x: $(OBJJS)
25: # hlk -x -o $@ $(OBJJS) -l libc.a libgmu.a
26:
27: slround.x: $(OBJJS2)
28: hlk -x -o $@ $(OBJJS2) -l libc.a libgmu.a libocsa.a
29:
30: #roundm.o: round.c
31: # gcc $(CC_OPTION) -o $@ -c <-DHAVEMAIN
32:
33: %.o: %.c
34: gcc $(CC_OPTION) -o $@ -c $<

```

▶D&G CGA SYSTEMに関する柴田氏の意見は、そのとおりだと思います。私の思っていたことと同じです。これからもマンマシンインタフェースを重視したツールを作ってください。特にCAD (モデラ) 関係をお願いします。 安丸 信吾(26) 神奈川県



面の順番を自動処理する

# ポリゴンソートフィルタ関数SortPoly()

Tan Akihiko 丹 明彦

SLASHのモデリング、特に面の順番制御は非常にやっかいな仕事です。  
SortPoly()関数は万能ではありませんが、ある程度までのデータなら自動  
的に正しい面順序に並べ替えてくれます。

## 概要

先月号の付録ディスクのSLASH開発キットに収録したポリゴンソートフィルタ関数SortPoly()のアルゴリズムを解説する。ポリゴンソートはポリゴナイザで正しい表示を行うために用いるものだが、SortPoly()関数は、SLASHの性格上、表示時でなくモデリング時に用いる前処理関数である。

## 構図

3次元コンピュータグラフィックスの流れを簡単におさらいしておく。

### 1) モデリング

形状データを構築する。ここでは、CADなどによる手動入力、プログラムによる自動生成を問わない。

### 2) 座標変換

座標系と視点と視線と物体の位置から、物体の座標に対して回転および平行移動の合成変換を行う。

### 3) 投影変換

物体をスクリーンに投影するための変換。視点から遠いものほど小さくして遠近感を出す透視投影と、これを行わない正投影とに分けられる。正投影は立体感が出ないので、SLASHでは透視投影のみを用いている。

なお、透視変換は変換後の座標に奥行き方向の座標(慣習的にz座標を用いる)を残すもので、透視投影は残さないもの。このあとの隠面消去とレンダリングのアルゴリズムによっては奥行き方向の座標は必要ないので、計算量の若干少ない透視投影を用いる場合もある。

### 4) クリッピング

余分な表示を防ぐために物体の画面からはみ出す部分を切り落とす。

### 5) 隠面消去

物体を構成するプリミティブが不透明な場合(SLASHのポリゴンは不透明である)、前後関係を正確に表現する必要がある。具体的には、奥にあるプリミティブが手前にあるプリミティブに隠れるようにする必要がある。

### 6) レンダリング

物体を表示する際に、陰影や光沢などの質感を与える。SLASHの場合、レンダリング手法として疑似ハイライトを加えたフラットシェーディングを用いている。

\* \* \*

1)~6)の工程は、必ずその順番に行わなくてはならないというわけではない。当然ながら、アルゴリズムは効率最優先で構築すべきである。

なお、CGシステムでレンダリングするといえ、2)~5)を行うことを指すことが多い。3次元CGの作品制作過程がモデリングとレンダリングに大別されるのは、レンダラがモデリング以外を全部やってしまうのが一般的なためである。

## 候補

さて、正しい表示のための手口あれこれを紹介する。もちろん、世の中に出回っている数々のアルゴリズムのなかのほんの一部である。

### 0) レイトレーシング

手法:ピクセルごとに、視線に最初に当たったプリミティブを描画する。

長所:座標変換や透視変換、クリッピングが不要(というよりアルゴリズムがそれらを内包している)。

短所:計算量が多い。リアルタイムはほぼ絶望的。

寸評:事実上静止画のみ。

レイトレーシングは半分冗談。以下は主にポリゴン用のアルゴリズムである。

### 1) ワイヤフレーム

手法:ポリゴンの辺のみを描く。

長所:隠面消去が必要ない。

短所:前後関係がややつかみにくい。

寸評:処理が軽い(X68000の場合)。動きが命。

### 2) 画家のアルゴリズム

手法:遠くのポリゴンから描く。

長所:わかりやすいアルゴリズム。

短所:透視変換後に(つまり毎フレームごとに)z座標をキーとしたデブスソートが必要。相貫体で破綻する。

寸評:画家はまず背景を描き、続いて近景を描き込んでいく。これが名前の由来。

### 3) 改良版画家のアルゴリズム

手法:ポリゴンを互いに交わる線で再分割したうえでデブスソートし、遠くから描く。

長所:相貫体もOK。

短所:ポリゴン数が極端に増える。

寸評:Z'sTRIPHONY DIGITALCRAFTで採用されたものだと思う。

### 4) zバッファ法

手法:ピクセルごとにz座標を算出し、zバッファを比較更新することによってピクセルごとにポリゴンの前後関係を確保する。

長所:事前のソートが不要。相貫体もOK。ハードウェア化しやすい。計算量も必要最小限。

短所:メモリを消費する。スキャンラインzバッファ法という対処法があるが、多少処理が複雑になる。

寸評:過去に3D特集で扱ったネタ。

### 5) スキャンライン法

手法:各スキャンラインでの前後関係の評価を幾何学的に真面目にやる。

長所:ポリゴンの幾何学的性質を完全に利用しているので確実かつ効率的に描画できる。

短所:計算がやや複雑。

寸評:D&GAは基本的にこのアルゴリズム。正統派。

\* \* \*



以下はSLASHで使われた技法である。

#### 6) バックフェーシング

手法：ポリゴンに表裏を設け、法線を設定しておき、透視変換後に表を向いているポリゴンのみ描画する。

長所：凸立体ではソートしなくても破綻しない。

短所：これだけに頼ると凹立体で破綻する可能性がある。複数の立体でも破綻する可能性がある。

#### 7) モデリング段階でのソート

手法：ポリゴンの位置関係によっては、視点によらず順序が決まる場合がある。

長所：描画時にソートする必要がない。

短所：ポリゴンの位置関係によっては、必ず破綻する組み合わせが存在する。

寸評：バックフェーシングが前提。今回紹介するSortPoly() 関数はこれを自動的に行う。

#### 8) ポリゴンマクロソート

手法：物体をいくつかのポリゴンマクロに分割し（各ポリゴンマクロ内のポリゴンは7)などによってソートされている）、透視変換後にそれぞれのポリゴンマクロの重心のz座標でデプスソートする。

長所：全ポリゴンを馬鹿正直にソートする必要がない。ソートが少なくて済む。

短所：モデリングや重心の位置決めにそれなりの配慮は必要。

\* \* \*

現実問題として、6)~8)を上手にやれば破綻しないし、破綻するとしても動かしていればほとんどわからない。バランスの取れた方法といえる。

まとめると、SLASH向けのポリゴンのソートは、3段階である。これらはいつソートするかで異なる。

- ・モデリング時（人手）
- ・実行時の初期設定（SortPoly() 関数）
- ・透視変換後（マクロソート）

SLASHでは前処理で吸収するのが基本となる。モデラを使って手で順番を決める方法は、面倒ではあるが、間違いがもっとも目立ちにくいように影響の少ないものから順番に並べていくことができる。SortPoly() 関数を使えば、ソートできるものは自動でソートする。どうしようもないものはマクロソートで解決するわけだ。

#### 9) モデルの性質を見る

手法：実現したいモデルの性質から、描く順番を決める。たとえば車は道のあとに描けばほぼ破綻しない。

長所：上手にやれば効率的になる。

短所：複雑なモデルでは制御しにくくなる。

寸評：これから連載のほうで頭を使いたい部分。

## 背景

我々はSLASHシステムを手に入れている。高速性を最優先して、本質的に必要でない判断された処理を徹底的に削ったポリゴナイザである。この削られた処理のなかにデプスソートがある。

デプスソートとは、透視変換後に視野に入る全ポリゴンを奥行き順にソートし、奥のポリゴンから描くことで前後関係を破綻させることなくシーンを描き上げるために用いられるソートである（前項の「画家のアルゴリズム」に用いられている）。

この方式は正攻法であるが、正直すぎるゆえSLASHには採用されていない。最大の欠点は、ポリゴン数が増えるほどソートに時間がかかってしまうということである。

SLASHはバックフェーシングを行うので、上手にモデリングすればまず問題ないことがわかる。さらにSLASHシステムが完成に近づくにつれ、人手でソートするのがけっこう大変だとわかる。自動的なソートを検討し始める。クイックソートを使うことにして、比較のための関数だけ用意すればいいのではないかと思ったが、これは失敗。そのうち、絶対ソートできない物体が多々あることははっきりしてきた。ポリ

ゴンマクロのソートが導入されたのもかなりあとになってからである。

そして、ひとつのポリゴンリストの中でできるだけ破綻しないようなソートを行うための関数SortPoly() が制作されたのであった。

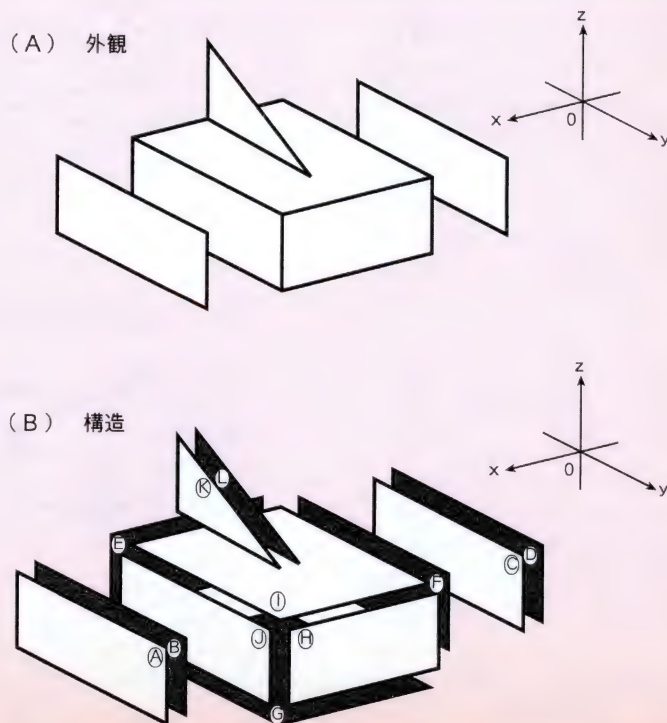
\* \* \*

ここで、今回の説明に用いる図形を紹介しておく。図1の簡易飛行機がそれだ。先月号の付録ディスクにもちょっと登場していたので覚えのある方もあるだろう。形状は見かけ（A）より少し複雑である（B）。尾翼と主翼は、1枚板のように見えるが、実は表裏2枚のポリゴンを張り合わせたものである。これは、SLASHがバックフェーシングによって裏向きのポリゴンを描かないことから必要なモデリングのテクニックであるが、ある意味では、この性質のおかげで、多くの場合において描画時に毎回デプスソートを行わなくて済むのである。そしてこの簡易飛行機もそのケースに含まれるのだ。

とはいうものの、なにも行わなければ表示時に破綻してしまうことに変わりはない。付録ディスクのサンプルプログラム「objtest.c」を（かなり）じっくりとご覧になれば、図1（B）のA~Lの順にポリゴンが定義されていることがわかる。

SLASHは、描画の際に物体の構造を基本的には考慮しない。つまり、ポリゴンが

図1 簡易飛行機





どのような配置になっているかには無頓着に、ポリゴンリストに定義されている順番に描く。その結果、図2 (A) のような事態が発生する。あとから描いたポリゴンは、それまでになが描かれていようと無条件に上描きするので、定義された順番のままでは前後関係が一部狂うのである。

そこでポリゴンソートを行うSortPoly()関数の登場となる。このソートは、前処理として用いる。つまりモデリング時または初期設定時に行うソートである。ソートの結果は静的に保持され、描画の間は変更する必要はない。図2 (B) をご覧いただきたい。これはSortPoly()の出力を調べて手作業でSLASHの描画をシミュレートしてみたものであるが、きちんとソートされていて、どの方向から見ても(図2 (A) で破綻していた方向から見ても)正しく表示されている。実際のプログラムを動かしても同様のことが確かめられるはずである。

## 戦略

ポリゴンソートの必要性和SortPoly()関数の効果を確かめたところで、プログラムの解説を始める。

基本戦略は、

- 1) ポリゴンの間に「順序」の概念を導入する
- 2) ポリゴンリストを「順序」に従ってソートする

というものである。「順序」とは、2枚のポリゴンが同時に可視となり、かつ画面上で重なる場合において、どちらが手前になるかということである。双方が表向きになるほうから見て、いつでも手前にあるものが優先順位が高い(ポリゴンリストの後方に配置される)。ポリゴンリストのなかでこの順序関係がきれいに成立していれば、このポリゴンリストは前処理だけで正しく描画できる物体ということになる。

だが、世の中いつでもうまくいかないもので、この方法では決してソートできないデータは存在する。典型的な例は、2枚のポリゴンがねじれの位置にあって、見る方向によってはお互いがお互いを隠し合うというもの。これはきわめてしばしば発生する。また、順序関係が循環している場合もソートできない(要するにグー・チョキ・パーを強さの順にはソートできないということ)。

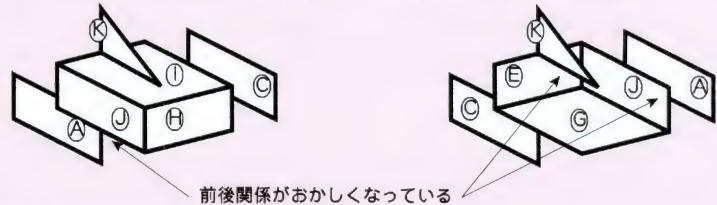
この、順序関係が必ずしも線形でないという性質のため、通常の線形ソート(バブルソートやクイックソートなど)ではソー

図2 SLASHの描画とポリゴンソートの必要性

SLASHはポリゴンをポリゴンリストに並んでいる順番に描画する。後ろ向きのポリゴンは描かない(バックフェーシング)。

### (A) ソート前

ポリゴンリストには A B C D E F G H I J K L の順で並んでいる。



B D E F G L は  
バックフェーシングにより  
除去(他も同様)

### (B) ソート後

ポリゴンリストは C B I K L J F A D E G H の順に並び替えられた。

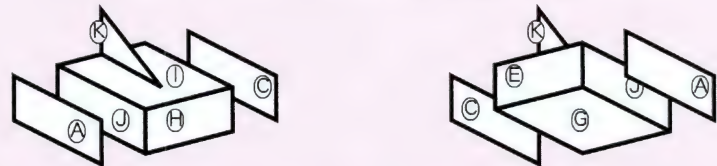
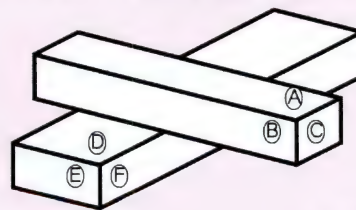


図3 ポリゴンの強弱関係

### (A) 2枚のポリゴンの強弱関係

2枚のポリゴン甲、乙に対する強弱関係の定義

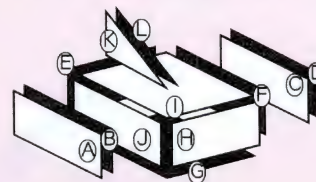
両者の表が見える位置から見たときに必ず甲が乙を覆い隠す……………甲>乙  
両者の表が見える位置から見たとき甲と乙が重ならない……………甲=乙  
見る位置によって甲が乙を覆い隠したり乙が甲を覆い隠したりする……………甲≠乙



A=B=C    D=E=F  
A>D    C>D    B>D  
A>E    C>E    B<E  
A>F    C>F    B≠F

≠の関係で結ばれるポリゴンの組がある場合、完全なポリゴンソートはできない。マクロソートで対処する必要がある。

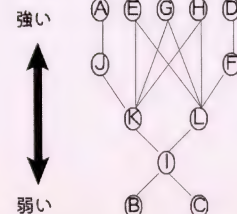
### (B) 簡易飛行機の場合



A>J>K>I>C  
D>F>L>I>B

H>K>C    E>K>C    G>K>C  
H>L>B    E>L>B    G>L>B

(強弱関係のグラフ)



≠の関係で結ばれるポリゴンの組がないので、完全なポリゴンソートが可能。



トできない。私は最初、線形ソートでうまくいくと思って見事にハマったのであった。

そこで、最終的に立てた戦略は次のようなものである。

- 1) ポリゴンの間に「強弱関係」の概念を導入する (図3)
- 2) ポリゴンの強弱関係を調べ、強弱関係行列を構築する (図4)
- 3) 強弱関係行列が望ましい形になるように、掃き出し法の要領でポリゴンリストを並べ替える (図5)

これで簡易飛行機のソートはうまくいっている。それでは具体的なアルゴリズムについて解説する。

## ■ 算法(1)強弱関係

まず、前項で定義なしに用いた「強弱関係」の概念を定義しておく (図3 (A))。これは2枚のポリゴンの間に成立する関係で、2枚のポリゴン甲と乙が画面上で重なって表示されるときに必要な。というよりも、当然のことながらポリゴンソートの必要性が出てくるのはこうした場面だけである。単純なデプスソートが無駄なのは、この事実を無視してソートしようとしているからということもできる。

で、その場合にポリゴン甲が必ずポリゴン乙を隠す、という関係が成立した場合に、ポリゴン甲はポリゴン乙より強い、と定義する。いうまでもないが、このような関係を導入できるのは、SLASHがバックフェーシングを行うからである。さもなくば、ある方向から見てポリゴン甲がポリゴン乙を覆い隠しても、それと正反対の方向から見ればポリゴン乙がポリゴン甲を覆い隠すことになり、強弱関係は決して成立しない。

図3 (A) をご覧いただきたい。ポリゴンAとポリゴンDの間には、完全な強弱関係が成立している。どの方向から見てもポリゴンAがポリゴンDを覆い隠すことはあっても、その逆はありえない。

ポリゴンAとポリゴンB、それにポリゴンCの間には明確な強弱関係が成立しない。なぜならどの方向から見ても決して重なることがないからである。一般にこれは凸立体では必ず成立するので、バックフェーシングを行っている限り、凸立体に対してソートを行う必要はないのである。

問題はポリゴンBとポリゴンFである。いわゆるねじれの位置にある。図の状態ではポリゴンBがポリゴンFを覆い隠しているので、ポリゴンBのほうが強そうではあるが、下のほうから見れば逆にポリゴンF

がポリゴンBを覆い隠している。実際の話、ポリゴンリストのなかにこうした関係のポリゴンの組がひとつでもあれば、ポリゴンソートは不可能である。

この場合は、2本の棒を別のポリゴンリストに定義し、2つのポリゴンリストをひとつのポリゴンマクロとして登録するしかない。そうすれば、SLASHは表示時にマクロソートを行うので、上から見たときにはポリゴンBが優先し、下から見たときにはポリゴンFが優先するようになる。

図3 (B) は、簡易飛行機の各ポリゴンの強弱関係である。こちらは、ソートを阻害するようなポリゴンの組が存在しないので、ポリゴンソートを行うことにより、ひとつのポリゴンマクロでも表示が破綻せずにすむ。強弱関係のグラフはおまけである。グ

ラフの下にあるものから順に描いていけば正しく表示できるはず。そういえば、開発中にはグラフをたどって順番を決定するというアルゴリズムも検討したっけ。

\* \* \*

2枚のポリゴンの強弱関係を求める算法の考え方を図4 (C) に示した。ベクトルの外積や内積を駆使すれば、比較的単純に求めることが可能である。最初は仮想的な視点を設定してそこからの距離を求めて……とやっていたのだが、それは頭の悪いやり方というものだ。

具体的な実装については先月号の付録ディスクのsortpoly.cのcompare() 関数およびprecompare() 関数を読んでいただきたい。これはSortPoly() 関数の下請け関数である。ポリゴンに三角形と四角形があるた

図4 ポリゴンの強弱関係行列

(A) 強弱関係行列

j	0	1	2	3	4	5	6	7	8	9	10	11
i	A	B	C	D	E	F	G	H	I	J	K	L
0 (A)	.	.	1	.	.	.	.	.	1	1	.	.
1 (B)	.	.	.	.	.	.	.	.	.	.	.	.
2 (C)	.	.	.	.	.	.	.	.	.	.	.	.
3 (D)	.	1	.	.	.	1	.	.	.	.	1	.
4 (E)	.	1	1	.	.	.	.	.	.	1	1	.
5 (F)	.	1	.	.	.	.	.	.	.	.	1	.
6 (G)	.	1	1	.	.	.	.	.	.	1	1	.
7 (H)	.	1	1	.	.	.	.	.	.	1	1	.
8 (I)	.	1	1	.	.	.	.	.	.	.	.	.
9 (J)	.	1	.	.	.	.	.	.	.	1	.	.
10 (K)	.	1	.	.	.	.	.	.	1	.	.	.
11 (L)	.	1	.	.	.	.	.	.	1	.	.	.

(i, j) 要素の意味  
= 1 ……ポリゴン i がポリゴン j を隠しうる  
= . ……それ以外

ポリゴン i > ポリゴン j の場合……

(i, j) 要素 = 1, (j, i) 要素 = 0

ポリゴン i = ポリゴン j の場合……

(i, j) 要素 = (j, i) 要素 = 0

(B) ソートされた状態

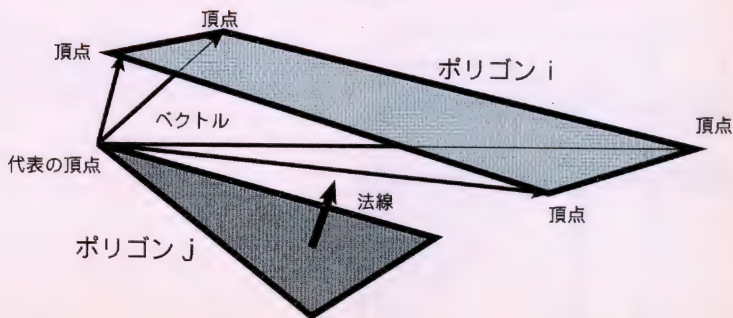
j	0	1	2	3	4	5	6	7	8	9	10	11
i	C	B	I	K	L	J	F	A	D	E	G	H
0 (C)	.	.	.	.	.	.	.	.	.	.	.	.
1 (B)	1	.	.	.	.	.	.	.	.	.	.	.
2 (I)	1	1	.	.	.	.	.	.	.	.	.	.
3 (K)	1	1	1	.	.	.	.	.	.	.	.	.
4 (L)	1	1	1	1	.	.	.	.	.	.	.	.
5 (J)	1	1	1	1	1	.	.	.	.	.	.	.
6 (F)	1	1	1	1	1	1	.	.	.	.	.	.
7 (A)	1	1	1	1	1	1	1	.	.	.	.	.
8 (D)	1	1	1	1	1	1	1	1	.	.	.	.
9 (E)	1	1	1	1	1	1	1	1	1	.	.	.
10 (G)	1	1	1	1	1	1	1	1	1	1	.	.
11 (H)	1	1	1	1	1	1	1	1	1	1	1	.

□ i > j (比較相手の優先順位が低い)  
■ i < j (比較相手の優先順位が高い)

ソートされた状態においては、  
i > j の領域に1が集まる。

自分より弱いポリゴンは必ず自分より番号が若い (優先順位が低い) という関係が常に保持されている。

(C) 強弱関係行列の要素を決定する算法



ポリゴン j 上の任意の点 (頂点のひとつを代表として選ぶ) からポリゴン i の各頂点へ向かうベクトルがすべてポリゴン j の法線と同方向を向いているならば、ポリゴン i はポリゴン j より強い。



め、多少場合分けがあるほかは比較的すっきりしたコーディングになっていると思う。

## ■ 算 法 ( 2 ) 強 弱 関 係 行 列

さて、強弱関係は把握したが、それをキーとして線形ソートしてもうまうまはいかない。さんざん考えた末に強弱関係行列を作ることになった。

強弱関係行列は (ポリゴン数) × (ポリゴン数) のテーブルで、各要素はあるポリゴンと別のポリゴンとの強弱関係を表現している。図 4 (A) は、簡易飛行機を作成した直後のポリゴンリストについて強弱関係行列を作ってみたものである。

さて、理想的にソートされたポリゴンリストの強弱関係行列はどのような形になるのであろうか？ それが図 3 (B) である。行列の対角要素を境界として、いわゆる下三角行列になっている。これはすなわち、

「どのポリゴンも、自分より先に描かれたポリゴンよりも優先順位が低いということがない」ことを意味する。これさえ保証されていれば、描画の際に前後関係が破綻することはありえない。

\* \* \*

さて、強弱関係行列の配列を理想的にするための算法である。

まず、ソートとは、リストのなかの要素を交換する作業の繰り返しとなる。ポリゴンリストの 2 つの要素を交換するとは、強弱関係行列に対してどういう操作を行うことに相当するのだろうか？ 答えは、行列の行と列を同時に交換することである。ポリゴンリストの i 番目の要素と j 番目の要素を交換することは、強弱関係の i 行と j 行、i 列と j 列を同時に交換することに相当する。

それでは、強弱関係を下三角行列にするために交換する行と列を選択するアルゴリズムとはどういうものか。これは関数 sor

tpoly.c 中の getnextpair() の仕事である。行列の掃き出し法にちょっとだけ似ている。

処理は優先順位の高いほうから行う (図 5 (A))。強いポリゴンのはうが検索しやすいためである。候補のうち、いちばん強いポリゴンを探し、ポリゴンリストの後ろのほうに移動する (図 5 (B))。強弱関係行列はそれを反映するように行および列を入れ替える。

## ■ 欠 点

あくまで、前処理関数であり、リアルタイム動作中に使うものではないと割り切って大急ぎで作った関数だから、細かい詰めはさぼっている。欠点も多い。あまり改良するつもりはない。

その 1

マトリクスの各要素の値は 0 か 1 しか取らないので、各 1 バイトというのはメモリの的には無駄が多い。ポリゴン数の 2 乗なので、たとえば 1000 枚のポリゴンに対しては 1M バイトのメモリを消費する。

解決案 1

マトリクスをビット単位で格納するだけで、メモリ消費量は 1/8 になる。

解決案 2

かなり疎な行列なので、節約の方法はあるに違いない。

その 2

あまり効率的なアルゴリズムとはいえない。基本的に掃き出し法に似た方法をとっているため、計算量のオーダーはポリゴン数の 3 乗程度になる。

その 3

SLPOLYGON 構造体の内容をソートの毎ステップでまるごと入れ替えている。

解決案

ポインタ参照を使って最後にまとめて入れ替えるようにすべき。

\* \* \*

以上のような問題点と対処法が考えられるが、どの程度効果的かは実装して試してみなければわからないところがある。まったく違うアルゴリズムというのも検討する価値はあろう。

SLASH で扱う物体には絶対にちゃんとした面の順番にできないものが存在する。こういったものに対して犠牲にするものを最小限に抑えた順番入れ替えアルゴリズムが求められているのだ。今回の関数はアプローチの一例にすぎない。よりよい関数の実現にぜひ挑戦してほしい。

図 5 簡易飛行機のポリゴンソート

### (A) ソートの進行

		0	1	2	3	4	5	6	7	8	9	10	11
初期状態		A	B	C	D	E	F	G	H	I	J	K	L
11 ⇔ 7		A	B	C	D	E	F	G	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
10 ⇔ 6		A	B	C	D	E	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
9 ⇔ 4		A	B	C	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
8 ⇔ 3		A	B	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
7 ⇔ 0		<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
6 ⇔ 5		<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
5 ⇔ 4		<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
4 ⇔ 0		<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
3 ⇔ 0		<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
2 ⇔ 0		<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
1 ⇔ 1		<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>
最終状態		C	B	I	K	L	J	F	A	D	E	G	H

優先度の高い順に決定する。  
優先度の高いポリゴンとは、  
あとに描画されるポリゴン、  
すなわち番号の大きいポリゴンである。

### (B) 強弱関係行列の行と列の交換

j	0	1	2	3	4	5	6	7	8	9	10	11
i	0	K										
1	B											
2	C											
3	I											
4	L											
5	J											
6	F											
7	A											
8	D											
9	E											
10	G											
11	H											

4 番目のポリゴンまで決定したときの様子。  
3 番目のポリゴン決定するために比較を行い、  
第 0 行・列と第 3 行・列を交換することに決定している。

処理の順序

処理の順序



基礎からのSLASH

# とりあえず三角錐を回してみる

Yamada Junji 山田 純二

驚異のポリゴナイザSLASH。その凄まじさに多くの読者の心が揺さぶられたようです。しかし、心揺り動かされてなにかを作ろうと思っても、システムを理解しなくてはなにもできません。まず三角錐を回してみましょう。

そろそろ10月号のアンケートが、返送され始めてきました。暇を見つけて……なんてことはなく、最近では、ハガキが来ると仕事を放り出してアンケートハガキを読みふけています。「SLASHシステムってすごいんだ」と感じた人「やっぱりねえ……」と感じた人、さまざまな反響が返ってきましたが、いちばん多かったのは「期待しています」というものでした。

でもとりあえず見ただけであきらめてしまう人が多かったのは少し残念です。確かに敷居は高いし、解説も十分とはいえなかったかもしれませんが。あの説明で理解できる人もいれば、そうでない人もいるのは当たり前。よほどの経験者でなければ、概要さえつかむのも難しかったでしょう。

しかし、せっかくすごいシステムが発表されたのに指をくわえて見ているだけでは、もったいないですよ。本当に。

今回は、せめて、ソースリストをアセンブルできるレベルのユーザーでも、自分の手でSLASHに触れることができるようなサンプルを用意し、解説をしてみます。タイトルにあるとおり、まずは三角錐を回すことから始めましょう。

## 画面表示の基本

まず、SLASHではどのように画面表示を行っているかを解説していきます。速度重視のため、SLASHでは最大256×256ドットの範囲にのみ描画を行います。これは、実画面が512×512ドットだろうと、1024×1024ドットだろうと関係ありません。つまり、画面に対する操作は、SETWPLNで設定されたアドレスを画面の左上アドレスとし、256×256ドットの範囲でクリッピングが自動的に行われます(図1)。

ここまでに関わるコールには、

### • SETWPLN

描画アドレスの設定

### • SETWINDOW

描画ウィンドウの範囲を設定

### • SETWINDOWCENTER

描画ウィンドウ左上を(0,0)とした座標で描画ウィンドウ中央を設定の3つがあり、それぞれの設定値の範囲は図2のようになります。

表示ルーチンではグラフィックを2ページ使うか、画面モードを256×256ドット(実画面512×512ドット)にして、左(0,0)-(255,255)と右(256,0)-(511,255)のエリアに、3Dオブジェクトを交互に消去、描画を行っていくことになります。実際には、使用ページ数を少なくするためにも、画面モード256×256ドット、実画面512×512ドットモードを使い、左右をグラフィックスクロールレジスタを切り替えて、疑似的に1ページ2画面として使用するのが、一般的な方法でしょう(図3)。

具体的には、

描画&消去アドレスの決定



3Dオブジェクトの消去

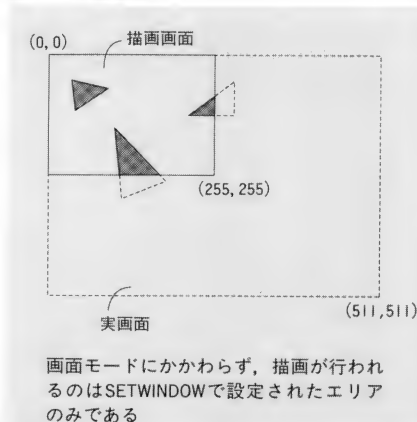


3Dオブジェクトの描画



画面切り替え

図1 描画範囲



のプロセスを踏んでいけばいいのです。

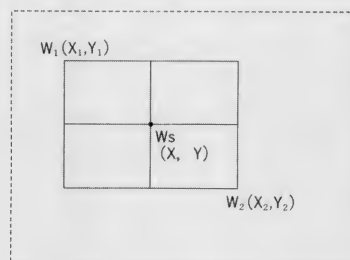
このようにいちいち交互に描画エリアを設定し、画面を切り替えて表示するのは、消去、描画プロセスで画面のちらつきを抑えるためです。同一画面で消去、描画を行うと処理速度の関係上、どうしてもちらつきが出てしまいます。

そこで、2つの描画エリアを用意し、左ページを表示している間に右ページの物体を消去、描画。描画が終わった段階で左右ページの切り替えを行います。切り替えが終わったら、今度は左ページの物体を消去、描画し……というふうに繰り返していきます。ちらつきの原因である消去、描画のプロセスを画面から見えないところで行うことによって、ちらつきを抑えることができるのです。

すでに、ここまでの説明でかなりの人は、「めんどくせ〜」と思ったり「消去&描画プロセスぐらいシステムでサポートしろよ」、ともいいたいかもしれません。しかし、逆にいえば、それだけユーザーの意思が反映できるのです。

たとえば、512×512ドットモード、256色

図2 描画エリアパラメータ設定範囲



描画アドレス (65536色モード)

$WAD = C00000H + X_1 \times 2 + Y_1 \times 400H$

描画ウィンドウ範囲

$Wx = X_2 - X_1 \quad (16 \leq Wx \leq 255)$

$Wy = Y_2 - Y_1 \quad (16 \leq Wy \leq 255)$

描画ウィンドウ中央

$X = Wx / 2$

$Y = Wy / 2$



モードを使い、グラフィックページ0,1を切り替えながら表示することによって、4分割マルチ画面モードなんてこともできます。このあたりをシステムで固定化されてしまうと、かえって4分割マルチ画面モードなどの応用をシステムをだましながら実現しなくてはならなくなり、処理速度の低下を招きかねません。

## 表示ルーチンの作成

それでは、これからSLASHで3D物体を表示するためのルーチンを作成していきます。ここで、10月号秋祭りPRO-68KのSION IVで使われていた、以下の2つのファイルが必要となります。

WORK.H……インデックスラベルが記述されたヘッダファイル

COLOR.S……カラーテーブル

これらは、サンプルプログラムをアセンブルするときに、必ず同じディレクトリに置いてくださいね。

それでは、表示ルーチンに取り掛かります。ここで必要なコールは、

### • SETWPLN

描画アドレスの設定

### • SETCPLN

消去アドレスの設定

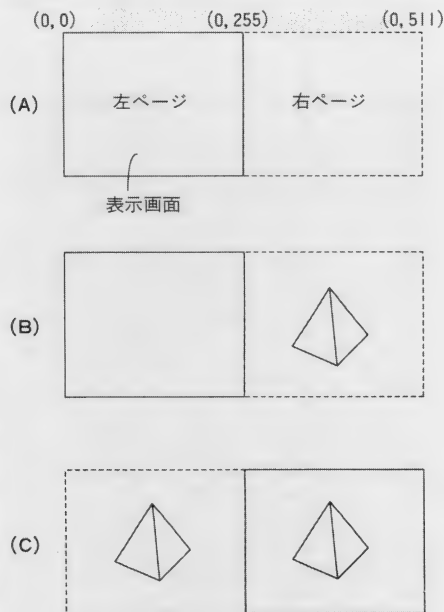
### • CLEARBOX

ミニマックスワークに格納された座標に従って画面消去を行う

### • TRANSLATER

3D物体の頂点データを2D座標に変換す

図3 画面切り替えの様子



まず、画面モード256×256ドット  
実画面512×512ドットに設定  
(0,0) - (255,255)を左ページ  
(256,0) - (511,255)を右ページとする

左ページを表示画面にし (グラフィックス  
クロールレジスタに(0,0)を設定) 右ページ  
の物体を消去し新しく描画する

右ページを表示画面にし (グラフィックス  
クロールレジスタに(0,256)を設定) 左ページ  
の物体を消去し新しく描画する

以下B, Cを繰り返す

る

### • DRAWPOLY

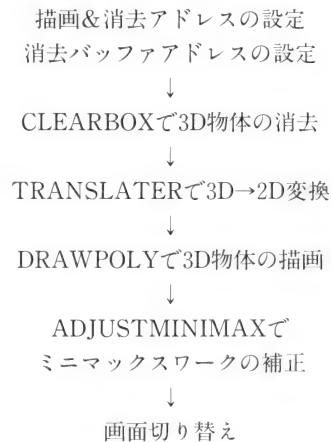
ポリゴン描画

### • ADJUSTMINIMAX

ミニマックスワークの補正

以上の6つです。

まず、簡単に処理の流れを見ていくと、



以上の手順で表示を行います。ここで気をつけてもらいたいのが、それぞれのコールに必要なワークエリアと、そのワークエリアのデータ構造です (表1)。

そして、いちばんの問題となるのが、物体消去に必要なミニマックスワークエリアの扱いです。左右ページを切り替えながら表示を行うわけですから、当然ミニマックスワークも2つ必要になります。左ページ的时候にはミニマックスワーク0を使い、右ページ的时候にはミニマックスワーク1を使うようにするのです。これを切り替えるタイミングは、処理の流れの先頭にある

とおり描画&消去アドレスの設定と一緒にを行います。

あとは、画面表示の基本のところの説明したとおり、アドレスの設定が行われたらCLEARBOXで物体消去を行い、次に新しい物体の描画を行えばいいのです。

では、物体の描画プロセスでどのようなワークの流れが起こっているか、どれだけのワークが必要となるか見てみましょう。

まず、3D物体を描画するために3DデータをTRANSLATERをコールして2D座標に変換します。このときには、以下のワークが必要となります。

• A6……3Dパラメータワーク

物体のX,Y,Z座標, PITCH, HEAD, BANKの回転角度, シューディング用のパラメータを合わせた16バイト

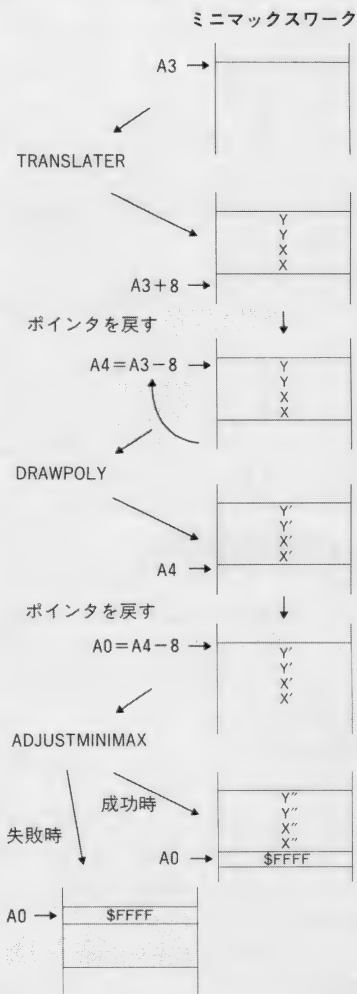
• A5……ポイントリスト

3D物体の頂点データ。頂点数×12+2バイト

• A4……トランスレートワーク

頂点ごとの2D変換情報。頂点数×32バイト

図4 ミニマックスワークの流れ





ト

・A3……ミニマックスワーク

物体数×8+2バイト

そして、コールが終わったときにA3レジスタが+8バイトされます。

次にDRAWPOLYを呼び出すときに、TRANSLATERで書き込まれたミニマックスワークを参照するので、A3レジスタを、

lea.l -8(A3),A3

としてポインタを戻しておく必要があります。そして、ポリゴンリスト先頭アドレスをA6レジスタ、TRANSLATERで2D変換したトランスレートワークをA5レジスタ、戻したミニマックスワークをA4レジスタにセットして、DRAWPOLYをコールします。

いよいよ、最後に呼び出すのがADJUSTMINIMAXです。ここで必要となるミニマックスワークは、DRAWPOLYで書き換えられたものですから、先ほどと同じようにポインタを戻してからADJUSTMINIMAXを呼び出してください。

言葉だけでは理解しづらいでしょうから、問題となるミニマックスワークのポインタの流れを、図4に書いておきましたので参照してください。

概要の説明が終わったところで、待望(?)の三角錐を回してみます。必要なものは、リスト1~3です。

まず、リスト2のOBJ.Sを、

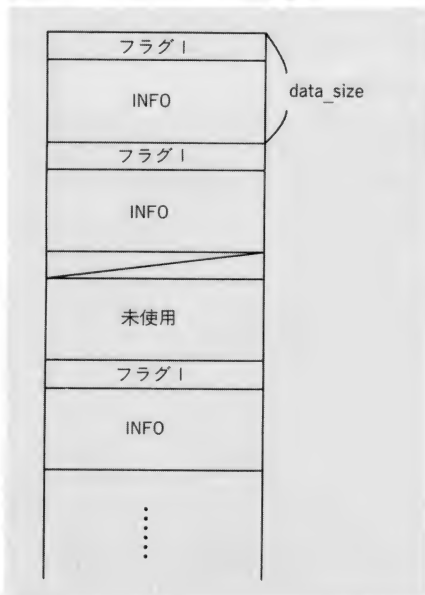
AS OBJ.S

とアセンブルし、

次にリスト3のTRLS.Sを、

AS TRLS

図5 キャラクタワークの構造



としてアセンブルし、

LK TRI OBJ SLASHLIB.A

としてリンクしてください。あとは、

TRI

と打ち込むだけで実行できます。ひととおりくるくる回る三角錐を観賞したあとは、ちょっとリスト3を見てください。最初のほうに、まだ解説していないコールが2つほど見つかりましたね。

#### ・SETCCL

CLEARBOXでクリアするときの消去カラーの設定

#### ・ADDNORM

ポリゴンのシェーディング情報、面法線を自動計算し、ポリゴンリストに書き加える

これら2つのコールは、特に説明の必要はないでしょう。SETCCLは、通常0にしておき、ADDNORMは、使いたい3D物体の数だけコールしておくだけです。

### 複数物体の制御

リスト2,3を打ち込んだ人は、すでにお気づきでしょうが、すでにリスト2の物体定義ルーチン(object\_put)は複数の物体定義に対応しています。ついでにSION IVでも使っているソートルーチンを組み込んであ

表1 各ワークエリアのデータ構造

#### 【パラメータ】 -----PARAMETER

```
00 x.w 物体X座標
02 y.w 物体Y座標
04 z.w 物体Z座標
06 pitch.w PITCH(X軸回りの回転角度)
08 head.w HEAD(Y軸回りの回転角度)
10 bank.w BANK(Z軸回りの回転角度)
12 α.w 光源緯度(0~31)
14 β.w 光源経度(0~63)
```

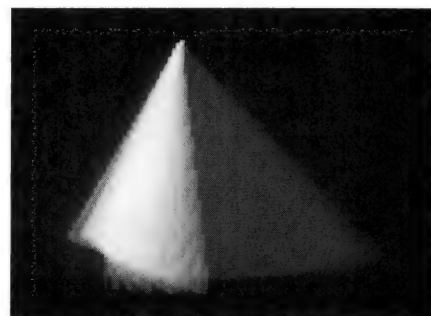
#### 【ポリゴンリスト】 -----POLY. LIST

n.w 面数  
ポリゴン1枚につき16ワード。以降は各ポリゴンヘッドからのオフセット値で示す。

```
[
00 type.w 識別子
           0:三角形 1:四角形
           2:線分 3:ポイント
02 point1.w 頂点1
           (ポイントリストの番号0~2047)
04 point2.w 頂点2
06 point3.w 頂点3
08 point4.w 頂点4
10 α.w 法線緯度(-1, 0~31)
12 β.w 法線経度(0~63)
14 color.l カラーテーブルアドレス、
           あるいはパレットコード
16~31 リザーブエリア
]
```

#### 【ポイントリスト】 -----POINT LIST

n.w 頂点数  
ポイント1点につき6ワード。以降は各ポイント



回る三角錐

ります。なお、このソートルーチンは、バブルソートという(バカソートともいう)かなりバカなアルゴリズムです。あくまで参考程度にしてください。

さて、複数物体に対応させるのは比較的簡単です。要するに物体ごとにある程度の大きさのワークエリアをもたせ、3Dパラメータワーク、3D物体頂点データリスト先頭アドレス、ポリゴンリスト先頭アドレスを格納すればいいのです(図5)。あとは、インデックスつきアドレッシングでそれぞれの値を設定するだけです。

とりあえずサンプルのリスト4です。

AS /D GAME.S

LK GAME OBJ SLASHLIB.A

としてアセンブルしてください。キー操作は、8,2,4,6キーで上下左右の移動、F10キ

ヘッドからのオフセット値で示す。

```
[
00 x.w 頂点X座標
02 y.w 頂点Y座標
04 z.w 頂点Z座標
]
```

#### 【トランスレートワークエリア】 -TRNS. WORK

ds.w 頂点数×8  
頂点ひとつにつき16バイト必要。2D変換後、作成されるデータは次のとおり。

```
[
00 x.w 3D計算後のX
02 y.w 3D計算後のY
04 z.w 3D計算後のZ
06 mx.w 2D変換後のX(成功時)
08 my.w 2D変換後のY(成功時)
10 z.w +04と同じ(成功時)
12 0.w
14 fig.w 0:2D変換成功, 1:失敗
]
```

#### 【ミニマックスワークエリア】 ---M.M. WORK

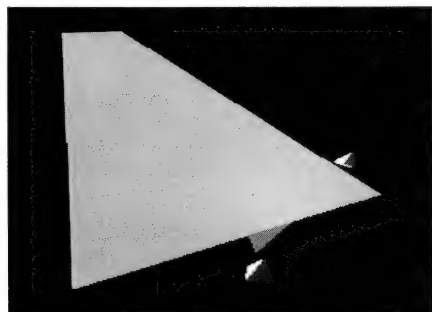
ds.w 物体数×4+1  
物体ひとつにつき8バイト+2バイト(エンドコード用)必要。ワークの内容は以下のとおり。

```
[
00 Ymin.w Y座標最小値
02 Ymax.w Y座標最大値
04 Xmin.w X座標最小値
06 Xmax.w X座標最大値
]
```

これを画面に対して有効な物体数分だけ用意する。

```
8n+0 $ffff.w エンドコード
```





複数物体制御

一で終了です。飛んでくる三角錐を画面中央に誘導しましょう。ちなみにスコアはカウントしていますが、表示されません。思いつき手抜きですが、どうしてもスコアが見たい人は、デバッグから起動し、メモリをダンプしてください。ラベル名atariが取った三角錐, apper\_kosuが登場した三角錐です。

## 弾けるオブジェクト

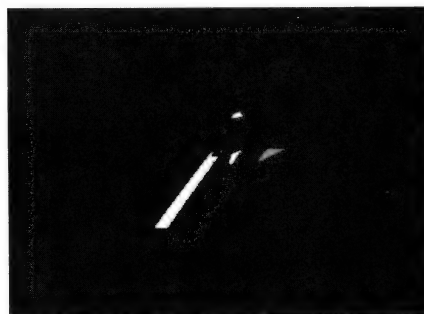
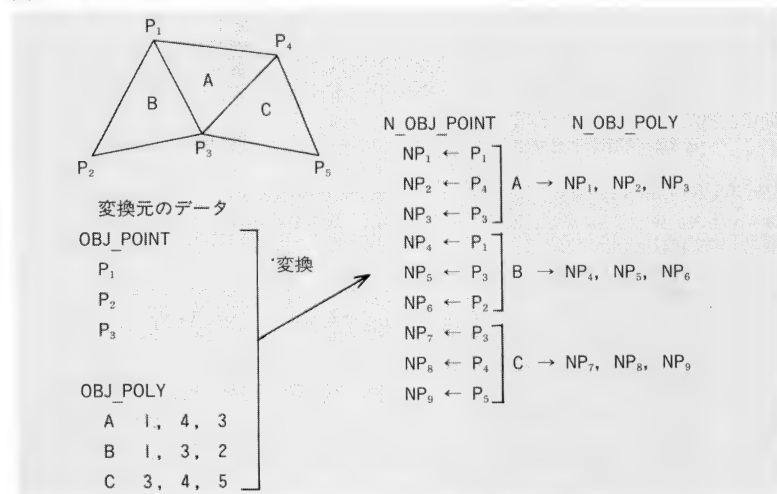
次は爆発です。ポリゴンモノのゲームではありがちなエフェクトですが、使いようによっては、かなりのインパクトを与えられます。スターブレードの要塞が崩れ落ちていくシーンや、メガCDのシルフィードでも戦艦の爆発などに使われていますね。

SLASHでこの弾けるポリゴン(?)を実現するためには、ポリゴン1枚1枚を動かした3D物体を自動生成する、という方法があります。つまり、それぞれのポリゴンに対する頂点をある一定の方向に移動してやればいいのです。そのために必要なことは2つあります。

1) 1頂点が必ず1ポリゴンに対応するようにする

普通、モデリングするときには、頂点の

図6 基本3D物体データの交換



弾けるオブジェクト

重複を避けるようにデータが作られます(同じ座標を変換するのは時間の無駄ですからね)。これを完全にポリゴンごとに頂点を対応させるのです(図6)。

## 2) 移動量テーブルの作成

これは、事前に各ポリゴンの飛んでいくベクトルを求め、その移動量をテーブル化してやるのです。で、この飛んでいくベクトルは重心のベクトルと同じである、と考えると話は簡単になります(図7)。要するに重心のベクトルのX,Y,Z成分は、ポリゴンの3頂点のX,Y,Z座標をそれぞれ足して3で割ったものになるのです。そうして、求めた重心ベクトルに倍率を掛け、分割数で割ったものが、1回に移動する移動量となります。

以上の2つのプロセスを通して出来上がった新しい3D物体の頂点データへ、移動テーブルに格納されている値をどんどん足して形状データを作成していけば、一応、それらしいものに仕上がります。

あとは、ポリゴンの位置によって移動量を変えたりとか(中央ほど移動量を大きくするなど)、各頂点ごとにバラバラな移動量を設定したりすると、いろいろ面白いことができるでしょう。

アセンブル方法は、

AS BOMB.S

LK BOMB OBJ SLASHLIB.A

でOKです。あと、形状データを変えるときには、ポリゴン数を250以下に抑えてください。でないとバッファをオーバーして暴走します。

## ワープデモ

今度はワープデモ(リスト6)。これも弾けるオブジェクトのように、3D物体を自動生成することで実現できますが、ちょっと趣向を変えてみましょう。なにを使うかというと、TRANSLATERで変換された2D座標バッファを使用するのです。

図7 移動量の計算方法

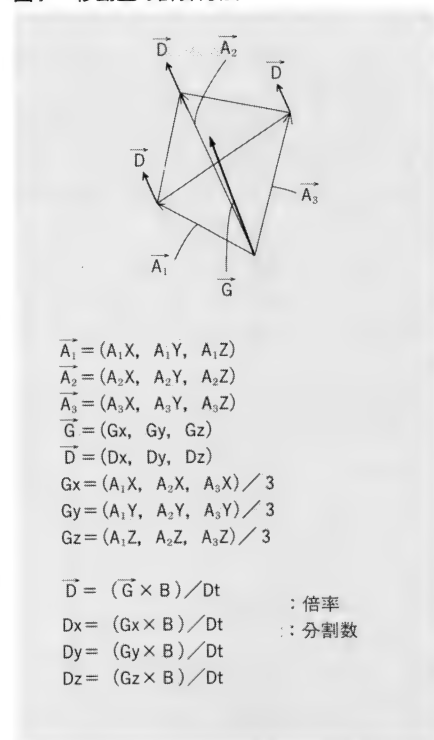
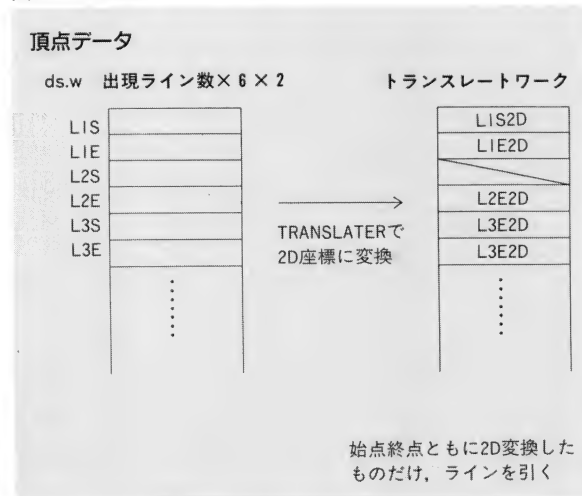


図8 2D変換ワークの利用





つまり、空間内に設定された星を、まず、ディスプレイ上へ投影し、その2D座標に従ってラインを描画するのです。気をつけるところは、始点終点ともに2D変換が正常に行われたもののみラインを描画する、という点です。あとは、Z座標に従ってカラーコードを変えたりとか、スピードによってラインの長さを変えるなどすれば、よりそれらしく見えることでしょう。

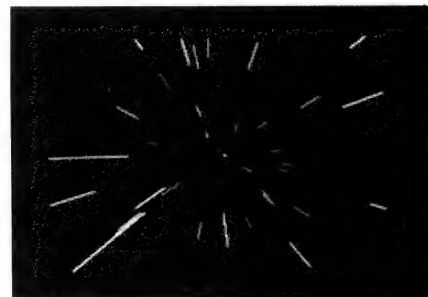
リスト6を、

AS STAR.S

LK STAR SLASHLIB.A

として実行ファイルを作成してください。キー操作は、8,2,4,6で上下左右の移動、XF1キーで加速、XF2キーで減速、F10キーで終了となっています。

では、皆さんがSLASHを活用できますように……。



ワープデモ

### リスト1

```
1: Tri      equ    0
2: Quad     equ    1
3: Line     equ    2
4:
5:
6: tri_point_data:
7:   dc.w    4
8:   dc.w    0,-15,0
9:   dc.w    0,15,-20
10:  dc.w    -20,15,20
11:  dc.w    20,15,20
12:
```

```
13: tri_poly_data:
14:   dc.w    4
15:   dc.w    Tri
16:   dc.w    0,1,2,0
17:   dc.w    0,0
18:   dc.l    TPL13
19:   ds.w    7
20:   dc.w    Tri
21:   dc.w    2,3,0,0
22:   dc.w    0,0
23:   dc.l    TPL13
24:   ds.w    7
```

```
25:   dc.w    Tri
26:   dc.w    0,3,1,0
27:   dc.w    0,0
28:   dc.l    TPL13
29:   ds.w    7
30:   dc.w    Tri
31:   dc.w    1,3,2,0
32:   dc.w    0,0
33:   dc.l    TPL13
34:   ds.w    7
35:
```

### リスト2

```
1:
2:
3: # 3Dオブジェクト定義ルーチン
4:
5:
6: .include      iocscall.mac
7: .include      doscall.mac
8: .include      work.h
9:
10: .xref  ADDNORM
11: .xref  SETCPLN
12: .xref  SETWPLN
13: .xref  TRANSLATER
14: .xref  TRANSLATER1
15: .xref  DRAWPOLY
16: .xref  CLEARBOX
17: .xref  ADJUSTMINIMAX
18: .xref  SETCCL
19:
20: .xdef  object_put
21: .xdef  object_sort
22: .xdef  page_set
23: .xdef  screen_change
24: .xdef  enemy_work
25: .xdef  erase_work0
26: .xdef  erase_work1
27:
28:
29: -----
30: # 画面切り替え
31:
32:
33: screen_change:
34:   move.w    gra_scroll,d0      *スクロールレジスタ設定
35:   lea.l     $e0000,a1
36:   move.w    d0,(a1)
37:   lea.l     2(a1),a1
38:   move.w    d0,(a1)+
39:   lea.l     2(a1),a1
40:   move.w    d0,(a1)+
41:   lea.l     2(a1),a1
42:   move.w    d0,(a1)+
43:   lea.l     2(a1),a1
44:   rts
45:
46: -----
47: # 3DオブジェクトをZ軸に従ってソートする
48:
49:
50: object_sort:
51:   lea.l     enemy_work,a1      *キャラクターワークアドレスの取り出し
52:   lea.l     sort_buf,a2        *ソートリスト格納バッファ
53:   move.w    #10-1,d7           *ワーク回数
54:   moveq.l   #0,d6              *ソート回数
55: objjs2:
56:   tst.w     (a1)               *キャラクターチェック
57:   beq       objjs_nextl
58:   *とりあえずソートバッファにワークアドレスを放り込む
59:   move.l     a1,(a2)+
60:   addq.w     #1,d6             *ソート回数++
61: objjs_nextl:
62:   lea.l     data_size(a1),a1
63:   dbf       d7,objjs2
64:
65:   subq.w     #2,d6
66:   bmi       objjs3            *物体定義回数<1ならソートしない
67:
68:   lea.l     sort_buf,a2        *ソートリスト格納バッファ
69: objjs5:
70:   movem.l    d6/a2,-(sp)
71:   move.l     (a2),a1           *最初のワークアドレスの取り出し
72:   move.l     a2,a4             *比較もとアドレスのコピー
73:   lea.l     4(a2),a2
74:   move.w     z(a1),d1          *Z座標の取り出し
75: objjs4:
76:   move.l     (a2),a3
77:   cmp.w     z(a3),d1
78:   bge       objjs6
79:
80:   move.l     a3,(a4)           *入れ替え
81:   move.l     a1,(a2)
82:   move.l     a3,a1
83:   move.w     z(a1),d1          *Z座標の取り出し
84: objjs6:
85:   lea.l     4(a2),a2           *次のソートリストアドレス計算
86:   dbf       d6,objjs4
87:
88:   movem.l    (sp)+,d6/a2
89:   lea.l     4(a2),a2
90:   dbf       d6,objjs5
91:   lea.l     4(a2),a2
```

```
92: objjs3:
93:   move.l     $ffffff,(a2)      *エンドコードの書き込み
94:   rts
95:
96: -----
97: # 描画アドレス決定と消去バッファセット
98:
99:
100: page_set:
101:   move.l     $c00000,a0
102:   moveq.l    #00,d0
103:   move.l     $erase_work0,era_addr0
104:
105:   eor.w      #1,g_flag
106:   bne        obp2
107:
108:   move.l     $c00200,a0
109:   move.w     #256,d0
110:   move.l     $erase_work1,era_addr0
111:
112: obp2:
113:   move.w     d0,gra_scroll      *スクロール座標をワークに格納
114:   jsr        SETCPLN           *消去アドレスの設定
115:   jsr        SETWPLN           *描画アドレスの設定
116:   move.l     era_addr0,a0       *定義されていたオブジェクトの消去
117:   jsr        CLEARBOX
118:   rts
119:
120: -----
121: # 3Dオブジェクト定義(グラフィック)
122:
123: object_put:
124:   lea.l     sort_buf,a0
125:   move.l     era_addr0,a3       *消去エリアバッファアドレスの取り出し
126:
127: obp3:
128:   tst.w     (a0)               *エンドコードチェック
129:   bmi       object_ret
130:   move.l     a0,-(sp)
131:
132:   move.l     (a0),a2           *ワークアドレスの取り出し
133:
134:   lea.l     2(a2),a6           *3Dパラメータポイント
135:   lea.l     point_buf,a5       *3D-2D変換格納バッファアドレス
136:   move.l     point_addr(a2),a4 *頂点データアドレスの取り出し
137:   move.l     a2,-(sp)
138:   jsr        TRANSLATER        *3D-2D変換
139:   move.l     (sp)+,a2
140:   cmp.w     #1,d0              *範囲外チェック
141:   bne        obp6
142:
143:   lea.l     -8(a3),a0
144:   bra        obp7
145:
146: obp6:
147:   move.l     poly_addr(a2),a6   *面データ格納アドレスの取り出し
148:   lea.l     point_buf,a5       *変換ポイントデータ格納アドレス
149:   lea.l     -8(a3),a4
150:   move.l     a4,-(sp)
151:   jsr        DRAWPOLY          *ポリゴン描画
152:   move.l     (sp)+,a0
153:
154: obp7:
155:   jsr        ADJUSTMINIMAX      *最大最小ワークの書き換え
156:   move.l     a0,a3
157:
158: obp5:
159:   move.l     (sp)+,a0
160:   lea.l     4(a0),a0
161:   bra        obp3
162:
163: object_ret:
164:   rts
165:
166: -----
167: # ワークエリア
168:
169: g_flag:      dc.w    00          * 描画ページ切り替えフラグ
170:
171: gra_scroll:  dc.w    0000        * グラフィックススクロール座標
172:
173:
174: era_addr0:   dc.l    $erase_work0
175:
176:
177: # ソートリスト格納バッファ
178:
179: sort_buf:    ds.l    200
180:
181:
182: -----
```



```

183: * 3D-2D変換用バッファ(1000個分)
184: * -----
185:
186: point_buf:
187:     ds.w    1000*8
188:
189: * -----
190: * 消去ワーク(オブジェクト100個分を用意)
191: * -----
192:
193: erase_work0:
194:     ds.w    4*100
195:     dc.w    $ffff
196:
197: erase_work1:
198:     * ページ0の消去バッファ
199:
200:
201:
202:
203:
204:
205:
206: * キャラクターワークエリア
207:
208: enemy_work:
209:     ds.w    data_size*10
210:
211:

```

```

198:     ds.w    4*100
199:     dc.w    $ffff
200:
201: dummy_all_erase:
202:     dc.w    0,255
203:     dc.w    0,255
204:     dc.w    $ffff
205:
206: * キャラクターワークエリア
207:
208: enemy_work:
209:     ds.w    data_size*10
210:
211:

```

### リスト3

```

1:
2: * 三角錐を回してみよう
3:
4:
5:
6:     .include    iocscall.mac
7:     .include    doscall.mac
8:     .include    work.h
9:
10:     .xref    ADDNORM
11:     .xref    SETCCL
12:
13:     .xref    object_put
14:     .xref    object_sort
15:     .xref    page_set
16:     .xref    screen_change
17:     .xref    enemy_work
18:     .xref    erase_work0
19:     .xref    erase_work1
20:
21: entry:
22:     move.w    #14,d1
23:     IOCS     _CRTMOD
24:     IOCS     _G_CLR_ON
25:     IOCS     _B_CUROFF
26:
27:     bsr      SUPER
28:
29:     moveq.l    #0,d0
30:     jsr      SETCCL
31:
32:     lea.l     tri_poly_data,a6
33:     jsr      ADDNORM
34:
35:     lea.l     enemy_work,a1
36:     move.w    #01,(a1)
37:     move.w    #0,x(a1)
38:     move.w    #0,y(a1)
39:     move.w    #100,z(a1)
40:     *ポリゴンリストデータアドレス
41:     move.l    #tri_poly_data,poly_addr(a1)
42:     *頂点データリストデータアドレス
43:     move.l    #tri_point_data,point_addr(a1)
44:
45:     move.w    #-1,erase_work0
46:

```

```

46:     move.w    #-1,erase_work1
47:
48:     move.w    #1000,d7
49: loop:
50:     lea.l     enemy_work,a1
51:     addq.w    #7,o_head(a1)
52:     move.l    d7/a1,-(sp)
53:     bsr      object_sort
54:     bsr      page_set
55:     bsr      object_put
56:     bsr      screen_change
57:     move.l    (sp)+,d7/a1
58:     dbf      d7,loop
59:
60:     bsr      USER
61:
62:     DOS      _EXIT
63:
64: * スーパーバイザモードへ
65:
66: SUPER:
67:     move.l    a1,-(sp)
68:     suba.l    a1,a1
69:     IOCS     _B_SUPER
70:     move.l    d0,sspbuff
71:     move.l    (sp)+,a1
72:     rts
73:
74: * ユーザーモードへ
75:
76: USER:
77:     move.l    a1,-(sp)
78:     move.l    sspbuff,a1
79:     IOCS     _B_SUPER
80:     move.l    (sp)+,a1
81:     rts
82:
83: sspbuff:
84:     dc.l     0
85:
86:
87:     .include    tri_data.s
88:     .include    color.s
89:
90:     .even

```

### リスト4

```

1:
2: * クルクル回る三角錐を取ってゲーム
3:
4:
5:     .include    iocscall.mac
6:     .include    doscall.mac
7:     .include    work.h
8:
9:     .xref    ADDNORM
10:    .xref    SETCCL
11:
12:    .xref    object_put
13:    .xref    object_sort
14:    .xref    page_set
15:    .xref    screen_change
16:    .xref    enemy_work
17:    .xref    erase_work0
18:    .xref    erase_work1
19:
20: FPACK macro callno
21:     dc.w    callno
22: endm
23:
24: __RAND equ $fe0e
25:
26: entry:
27:     move.w    #14,d1
28:     IOCS     _CRTMOD
29:     IOCS     _G_CLR_ON
30:     IOCS     _B_CUROFF
31:
32:     bsr      SUPER
33:
34:     moveq.l    #0,d0
35:     jsr      SETCCL
36:
37:     lea.l     tri_poly_data,a6
38:     lea.l     tri_point_data,a5
39:     jsr      ADDNORM
40:
41:     move.w    #-1,erase_work0
42:     move.w    #-1,erase_work1
43:     bsr      init_work
44:
45: loop:
46:     bsr      key_in
47:     bsr      appear_tri
48:     bsr      tri_main
49:
50:     bsr      object_sort
51:     bsr      page_set
52:     bsr      object_put
53:     bsr      screen_change
54:
55:     moveq.l    #0,d1
56:     IOCS     _BITSNS
57:     btat     #04,d0
58:     beq      loop
59:
60:     bsr      USER
61:
62:     DOS      _EXIT

```

```

63: * 三角錐を出現させる
64: * -----
65:
66: appear_tri:
67:     subq.w    #1,appear_cnt
68:     beq      apt2
69:     rts
70:
71: apt2:
72:     lea.l     enemy_work,a1
73:     moveq.l    #9,d1
74:
75:     tst.w     (a1)
76:     beq      apt4
77:     lea.l     data_size(a1),a1
78:     dbf      d1,apt3
79:
80:     move.w    #1,appear_cnt
81:     rts
82:
83: apt4:
84:     move.w    #200,appear_cnt
85:     move.w    #1,(a1)
86:     FPACK    __RAND
87:     and.w     #01ff,d0
88:     sub.w     #256,d0
89:     move.w    d0,x(a1)
90:     FPACK    __RAND
91:     and.w     #01ff,d0
92:     sub.w     #256,d0
93:     move.w    d0,y(a1)
94:     FPACK    __RAND
95:     and.w     #01ff,d0
96:     sub.w     #256,d0
97:     move.w    d0,z(a1)
98:     FPACK    __RAND
99:     and.w     #01ff,d0
100:    sub.w     #256,d0
101:    move.l     #tri_poly_data,poly_addr(a1)
102:    move.l     #tri_point_data,point_addr(a1)
103:    addq.w     #1,appear_kosu
104:    rts
105:
106: * -----
107: * 三角錐を移動させる
108: * -----
109:
110: tri_main:
111:     lea.l     enemy_work,a1
112:     moveq.l    #9,d7
113:
114:     trm2:
115:     move.l     d7,-(sp)
116:     tst.w     (a1)
117:     beq      trm3
118:     bsr      tri_move
119:     bsr      area_chk
120:
121:     trm3:
122:     lea.l     data_size(a1),a1
123:     move.l     (sp)+,d7
124:     dbf      d7,trm2
125:     rts
126:
127: * -----
128: * 移動ルーチン
129: * -----
130:
131: tri_move:
132:     addq.w     #8,o_head(a1)
133:     addq.w     #8,o_bank(a1)

```



```

125:      sub.w    #1,z(a1)
126:      move.w   x_id,d0
127:      sub.w    d0,x(a1)      *自機移動量を足す
128:      move.w   y_id,d0
129:      sub.w    d0,y(a1)
130:      rts
131:
132:  -----
133:  * 敵機チェック
134:  -----
135:
136:  area_chk:
137:      cmp.w    #40,z(a1)      *Z<40なら当たり判定チェック
138:      bge      arc2
139:
140:      cmp.w    #-50,x(a1)      * -50<x<50かつ-50<y<50
141:      ble      arc2            *なら当たり
142:      cmp.w    #50,x(a1)
143:      bge      arc2
144:      cmp.w    #-50,y(a1)
145:      ble      arc2
146:      cmp.w    #50,y(a1)
147:      bge      arc2
148:      addq.w   #1,atari
149:      clr.w    (a1)            *物体消去
150:  arc2:
151:      rts
152:
153:  -----
154:  * キー入力
155:  -----
156:
157:  key_in:
158:      moveq.l  #0,d2            *X方向の移動量
159:      moveq.l  #0,d3            *Y方向の移動量
160:      move.l   #8,d1            *4キーのチェック
161:      IOCS     _BITSNS
162:      btst     #7,d0
163:      beq      up_chk
164:      subq.w   #1,d2
165:
166:  up_chk:
167:      btst     #4,d0            *8キーのチェック
168:      beq      right_chk
169:      subq.w   #1,d3
170:
171:  right_chk:
172:      move.l   #9,d1            *6キーのチェック
173:      IOCS     _BITSNS
174:      btst     #1,d0
175:      beq      down_chk
176:      addq.w   #1,d2
177:
178:  down_chk:
179:      btst     #4,d0            *2キーのチェック
180:      beq      chk_out
181:      addq.w   #1,d3
182:
183:  chk_out:
184:      move.w   d2,x_id          *移動量の格納
185:      move.w   d3,y_id
186:      rts
187:
188:  -----
189:  * ワーク初期化
190:  -----

```

```

187:
188:  init_work:
189:      lea.l    enemy_work,a1
190:      moveq.l  #9,d1
191:
192:  iw2:
193:      clr.w    (a1)
194:      lea.l    data_size(a1),a1
195:      dbf      d1,iw2
196:      rts
197:
198:  * スローリザモードへ
199:
200:  SUPER:
201:      move.l   a1,-(sp)
202:      suba.l   a1,a1
203:      IOCS     _B_SUPER
204:      move.l   d0,asppbuf
205:      move.l   (sp)+,a1
206:      rts
207:
208:  * ユーザーモードへ
209:
210:  USER:
211:      move.l   a1,-(sp)
212:      movea.l  asppbuf,a1
213:      IOCS     _B_SUPER
214:      move.l   (sp)+,a1
215:      rts
216:
217:  asppbuf:
218:      dc.l     0
219:
220:  WAITDISP:
221:      lea.l    #e88001,a6
222:      moveq.l  #4,d7
223:
224:  waitdisp1:
225:      btst.b   d7,(a6)
226:      beq      waitdisp1      #now    returning
227:
228:  waitdisp2:
229:      btst.b   d7,(a6)
230:      bne      waitdisp2      #now    printing
231:      rts
232:
233:  * ワークエリア
234:
235:  appear_cnt:
236:      dc.w     0001
237:
238:  x_id:
239:      dc.w     0000
240:
241:  y_id:
242:      dc.w     0000
243:
244:  atari:
245:      dc.w     0000
246:
247:  appear_kosu:
248:      dc.w     0000
249:
250:      .include  tri_data.s
251:      .include  color.s
252:
253:  .even

```

## リスト5

```

1:
2:  * 弾けるオブジェクト
3:
4:      .include  iocscall.mac
5:      .include  doscall.mac
6:      .include  work.h
7:
8:      .xref    ADDNORM
9:      .xref    SETCCL
10:
11:      .xref    object_put
12:      .xref    object_sort
13:      .xref    page_set
14:      .xref    screen_change
15:      .xref    enemy_work
16:      .xref    erase_work0
17:      .xref    erase_work1
18:
19:  entry:
20:      move.w   #14,d1
21:      IOCS     _CRTMOD
22:      IOCS     _G_CLR_ON
23:      IOCS     _B_CUROFF
24:
25:      bar      SUPER
26:      moveq.l  #0,d0            *消去カラーの設定
27:      jsr      SETCCL
28:
29:      bar      point_poly_make *頂点&ポリゴンリストの作成
30:      bar      id_list_make   *移動量テーブルの作成
31:
32:      lea.l    n_poly_data,a6
33:      lea.l    n_point_data,a5
34:      jsr      ADDNORM
35:
36:      lea.l    enemy_work,a1
37:      move.w   #01,(a1)        *初期化データセット
38:      move.w   #0,x(a1)        *使用フラグ
39:      move.w   #0,y(a1)        *表示XY Z座標
40:      move.w   #4000,z(a1)
41:      move.w   #512,o_pitch(a1) *回転角
42:      move.w   #512,o_head(a1)
43:
44:      move.l   #n_poly_data,poly_addr(a1)
45:      move.l   #n_point_data,point_addr(a1)
46:
47:      move.w   #-1,erase_work0 *消去バッファの初期化
48:      move.w   #-1,erase_work1
49:
50:      move.w   #50,d7
51:
52:  loop:
53:      move.l   d7/a1,-(sp)
54:      bar      id_object_make
55:      jsr      object_sort
56:      jsr      page_set
57:      jsr      object_put
58:      jsr      screen_change
59:      move.l   (sp)+,d7/a1
60:      dbf      d7,loop
61:
62:  ret:

```

```

64:      bar      USER
65:      DOS      _EXIT
66:
67:  -----
68:  * 頂点リスト&ポリゴンリストの作成
69:  -----
70:
71:  point_poly_make:
72:      lea.l    tri_point_data+2,a1 *変換もとの頂点データリスト
73:      lea.l    tri_poly_data,a2   *変換もとのポリゴンデータリスト
74:
75:      lea.l    n_point_data+2,a3 *変換先の頂点データリスト
76:      lea.l    n_poly_data+2,a4   *変換先のポリゴンデータリスト
77:
78:      move.w   (a2)+,d7
79:      move.w   d7,-2(a4)
80:      subq.w   #1,d7
81:      moveq.l  #0,d1
82:
83:  ppm2:
84:      moveq.l  #2,d6
85:      move.w   (a2)+,d0
86:      add.w    d0,d6
87:      move.w   d0,(a4)+
88:
89:  ppm3:
90:      moveq.l  #0,d5
91:      move.w   (a2)+,d5
92:      add.l    d5,d5
93:      move.l   d5,d4
94:      add.l    d5,a1
95:      move.w   0(a1),(a3)+
96:      move.w   2(a1),(a3)+
97:      move.w   4(a1),(a3)+
98:      move.l   (sp)+,a1
99:
100:      move.w   d1,(a4)+
101:      addq.w   #1,d1
102:      dbf      d6,ppm3
103:
104:      eor.w    #1,d0
105:      add.w    d0,d0
106:      lea.l    4(a2),a2
107:      add.w    d0,a2
108:      lea.l    4(a4),a4
109:      add.w    d0,a4
110:      move.l   (a2),(a4)
111:      lea.l    18(a2),a2
112:      lea.l    18(a4),a4
113:      dbf      d7,ppm2
114:      move.w   d1,n_point_data
115:      rts
116:
117:  -----
118:  * 移動量リストの作成
119:  -----
120:
121:  id_list_make:
122:      lea.l    n_point_data+2,a1
123:      lea.l    n_poly_data,a2
124:      lea.l    n_id_data,a3
125:

```



```

126: move.w (a2)+,d7      *変換ポリゴン関数の取り出し
127: subq.w #1,d7
128: ilw2:
129: move.w (a2),d6      *識別子の取り出し
130: add.w #2,d6          *頂点個数の計算
131: moveq.l #0,d1        *X方向の移動量
132: moveq.l #0,d2        *Y方向の移動量
133: moveq.l #0,d3        *Z方向の移動量
134: ilw3:
135: add.w (a1)+,d1        *X方向の加算
136: add.w (a1)+,d2        *Y方向の加算
137: add.w (a1)+,d3        *Z方向の加算
138: dbf d6,ilw3
139: muls bind,d1          *(ベクトルX×倍率)/分割数
140: divs d_cnt,d1
141: move.w d1,(a3)+      *移動量の格納
142:
143: muls bind,d2          *(ベクトルY×倍率)/分割数
144: divs d_cnt,d2
145: move.w d2,(a3)+      *移動量の格納
146:
147: muls bind,d3          *(ベクトルZ×倍率)/分割数
148: divs d_cnt,d3
149: move.w d3,(a3)+      *移動量の格納
150: lea.l 32(a2),a2      *次のポリゴンリストへ
151: dbf d7,ilw2
152: rts
153:
154: -----
155: * オブジェクトの作成
156: -----
157:
158: id_object_make:
159: lea.l n_point_data+2,a1 *頂点リスト
160: lea.l n_poly_data,a2   *ポリゴンリスト
161: lea.l n_id_data,a3     *移動量リスト
162:
163: move.w (a2)+,d7      *変換ポリゴン関数の取り出し
164: subq.w #1,d7
165: ilw2:
166: move.w (a3)+,d1      *各移動量の取り出し
167: move.w (a3)+,d2
168: move.w (a3)+,d3
169: move.w (a2),d6      *識別子の取り出し
170: add.w #2,d6
171: ilw3:
172: add.w d1,(a1)+      *頂点座標に移動量を加算
173: add.w d2,(a1)+
174: add.w d3,(a1)+
175: dbf d6,ilw3
176: lea.l 32(a2),a2
177: dbf d7,ilw2
178: rts
179:
180: * スーパーバイザモードへ
181:
182: SUPER:

```

```

183: move.l a1,-(sp)
184: suba.l a1,a1
185: IOCS _B_SUPER
186: move.l d0,ssbuf
187: move.l (sp)+,a1
188: rts
189:
190: * ユーザーモードへ
191:
192: USER:
193: move.l a1,-(sp)
194: move.l ssbuf,a1
195: IOCS _B_SUPER
196: move.l (sp)+,a1
197: rts
198:
199: ssbuf:
200: dc.l 0
201:
202: WAITDISP:
203: lea.l $e88001,a6
204: moveq.l #4,d7
205: waitdisp1:
206: btst.b d7,(a6)
207: beq waitdisp1 *now returning
208: waitdisp2:
209: btst.b d7,(a6)
210: bne waitdisp2 *now printing
211: rts
212:
213: * ワーク
214:
215: bind:
216: dc.w 01 *倍率
217: d_cnt:
218: dc.w 20 *分割数
219:
220: * 変換ワーク
221:
222: n_point_data:
223: dc.w 00 *頂点数
224: ds.w 03*1000
225:
226: n_poly_data:
227: dc.w 00 *ポリゴン数
228: ds.w 16*1000
229:
230: n_id_data:
231: ds.w 03*1000 *移動量リスト
232:
233: .include tri_data.s
234: .include color.s
235:
236: .even
237:

```

## リスト6

```

1:
2: * ワープデモ?
3:
4: .include iocscall.mac
5: .include doscall.mac
6: .include work.h
7:
8: .xref TRANSLATER
9: .xref SETWPLN
10: .xref LINER2
11:
12: star_cnt equ 60
13:
14: FPACK macro callno
15: dc.w callno
16: endm
17:
18: __RAND equ $fe0e
19:
20: entry:
21: move.w #14,d1
22: IOCS _CRTMOD
23: IOCS _G_CLR_ON
24: IOCS _B_CUROFF
25:
26: bsr SUPER
27:
28: bsr star_init *ワーク初期化
29: loop:
30: bsr key_in *キー入力
31: bsr star_main *星の座標移動
32: bsr page_set *描画ページ決定
33: bsr star_era *消去
34: bsr star_put *描画
35: bsr screen_change *描画ページの切り替え
36:
37: moveq.l #80,d1 *f10キーチェック
38: IOCS _BITSNS
39: btst #04,d0
40: beq loop
41:
42: bsr USER
43:
44: DOS _EXIT
45:
46: -----
47: * キー入力
48: -----
49:
50: key_in:
51: moveq.l #0,d2 *X方向の移動量
52: moveq.l #0,d3 *Y方向の移動量
53: move.l #8,d1 *f1キーのチェック
54: IOCS _BITSNS
55: btst #7,d0
56: beq up_chk
57: subq.w #1,d2
58: up_chk:
59: btst #4,d0 *f8キーのチェック
60: beq right_chk
61: subq.w #1,d3
62: right_chk:
63: move.l #9,d1 *f6キーのチェック
64: IOCS _BITSNS
65: btst #1,d0
66: beq down_chk
67: addq.w #1,d2
68: down_chk:

```

```

69: btst #4,d0 *f2キーのチェック
70: beq chk_out
71: addq.w #1,d3
72: chk_out:
73: move.w d2,x_id *移動量の格納
74: move.w d3,y_id
75:
76: moveq.l #10,d1
77: IOCS _BITSNS
78:
79: btst #05,d0 *xf1キーのチェック
80: beq dchk
81: addq.w #1,speed
82: cmp.w #150,speed
83: ble chk2
84: move.w #150,speed
85: bra chk2
86: dchk:
87: btst #05,d0 *xf2キーのチェック
88: beq chk2
89: subq.w #1,speed
90: bne chk2
91: move.w #1,speed
92: chk2:
93: rts
94:
95: -----
96: * 星のメインルーチン
97: -----
98:
99: star_main:
100: lea.l star_para,a6 *背景の星を3D-2D変換
101: move.w #-500,4(a6)
102:
103: lea.l star_map_data+2,a0
104: moveq.l #star_cnt-1,d7
105: stm2:
106: move.w speed,d0
107: sub.w d0,4(a0) *移動
108: bpl stm3
109: add.w #2048,4(a0)
110: stm3:
111: move.w x_id,d0 *X方向の移動量を加算
112: add.w d0,d0
113: add.w d0,d0
114: move.w (a0),d1
115: sub.w d0,d1
116: cmp.w #512,d1
117: ble stm4
118: stm4:
119: sub.w #1024,d1
120: bra stm5
121: stm1:
122: cmp.w #-512,d1
123: bge stm5
124: add.w #1024,d1
125: stm5:
126: move.w d1,(a0)
127:
128: move.w y_id,d0 *Y方向の移動量を加算
129: add.w d0,d0
130: add.w d0,d0
131: add.w d0,d0
132: move.w 2(a0),d1
133: sub.w d0,d1
134: cmp.w #512,d1
135: ble stm6
136: sub.w #1024,d1

```



```

137:      bra      stm7
138: stm6:      cmp.w    #-512,d1
139:            bge     stm7
140:            add.w    #1024,d1
141:
142: stm7:      move.w   d1,2(a0)
143:
144:            move.w   (a0),6(a0)
145:            move.w   2(a0),8(a0)
146:            move.w   4(a0),10(a0)
147:            move.w   speed,d0
148:            add.w    d0,d0
149:            add.w    d0,10(a0)
150:
151:            lea.l    12(a0),a0
152:            dbf      d7,stm2
153:            rts
154:
155:
156: #
157: # 背景の星を描画
158: #
159:
160: star_put:
161:      lea.l    star_para,a6      #背景の星を3D-2D変換
162:      move.l   star_point_addr,a5
163:      lea.l    star_map_data,a4
164:      lea.l    star_back,a3
165:      jar      TRANSLATER
166:
167:      lea.l    star_grad_data,a2
168:      lea.l    star_map_data+2,a1
169:      move.l   star_point_addr,a0
170:      moveq.l  #star_cnt-1,d7
171:
172: st_p2:      move.l   d7,-(sp)
173:
174:            tst.w    14(a0)      #変換成功か?
175:            bne     st_p3
176:
177:            move.w   6(a0),d6      #2D座標の取り出し(始点)
178:            move.w   8(a0),d2
179:
180:            tst.w    14+16(a0)    #終点も変換成功か?
181:            bne     st_p3
182:
183:            move.w   6+16(a0),a4    #終点の取り出し
184:            move.w   8+16(a0),d7
185:            #2軸の座標によってカラーコードを変える
186:            move.w   4(a1),d3
187:            asr.w    #7,d3
188:            move.w   #15,d4
189:            sub.w    d3,d4
190:            add.w    d4,d4
191:            move.w   (a2,d4.w),d0
192:
193:            moveq.l  #0,d1      #参照点=0
194:            move.l   d1,a3
195:
196:            movem.l  a0-a2,-(sp)
197:            jar      LINER2
198:            movem.l  (sp)+,a0-a2
199:
200: st_p3:      lea.l    32(a0),a0
201:            lea.l    12(a1),a1
202:
203:            move.l   (sp)+,d7
204:            dbf      d7,st_p2
205:            rts
206:
207: #
208: # 背景の星を消去
209: #
210:
211: star_era:
212:      move.l   star_point_addr,a0
213:      moveq.l  #star_cnt-1,d7
214:
215: st_e2:      move.l   d7,-(sp)
216:
217:            tst.w    14(a0)      #変換成功か?
218:            bne     st_e3
219:
220:            move.w   6(a0),d6      #2D座標の取り出し
221:            move.w   8(a0),d2
222:            tst.w    14+16(a0)    #終点も変換成功か?
223:            bne     st_e3
224:
225:            move.w   6+16(a0),a4    #終点の取り出し
226:            move.w   8+16(a0),d7
227:
228:            moveq.l  #0,d1      #参照点=0
229:            move.l   d1,a3
230:
231:            moveq.l  #0,d0
232:            move.l   a0,-(sp)
233:            jar      LINER2
234:            movem.l  (sp)+,a0
235:
236: st_e3:      lea.l    32(a0),a0
237:            move.l   (sp)+,d7
238:            dbf      d7,st_e2
239:            rts
240:
241: #
242: # 背景の星の点データを定義する&ワークの初期化
243: #
244:
245: star_init:
246:      lea.l    star_map_data+2,a0
247:      moveq.l  #star_cnt-1,d7
248:
249: st_i2:      FPACK     _RAND
250:            and.w    #03ff,d0
251:            sub.w    #512,d0
252:            move.w   d0,(a0)+      #X座標セット
253:            move.w   d0,4(a0)
254:
255:            FPACK     _RAND
256:            and.w    #03ff,d0
257:            sub.w    #512,d0
258:            move.w   d0,(a0)+      #Y座標セット
259:            move.w   d0,4(a0)
260:
261:            FPACK     _RAND
262:            and.w    #07ff,d0
263:            move.w   d0,(a0)+      #Z座標セット
264:            lea.l    6(a0),a0
265:            dbf      d7,st_i2
266:

```

```

267:      lea.l    star_work0+14,a0    #ワークの初期化
268:      moveq.l  #star_cnt*2-1,d7
269:
270: st_i3:      move.w   #1,(a0)
271:            lea.l    16(a0),a0
272:            dbf      d7,st_i3
273:            lea.l    star_work1+14,a0
274:            moveq.l  #star_cnt*2-1,d7
275:
276: st_i4:      move.w   #1,(a0)
277:            lea.l    16(a0),a0
278:            dbf      d7,st_i4
279:            rts
280:
281: #-----
282: # 画面切り替え
283: #-----
284:
285: screen_change:
286:      move.w   gra_scroll,d0      #スクロールレジスタ設定
287:      lea.l    #00018,a1
288:      move.w   d0,(a1)+
289:      lea.l    2(a1),a1
290:      move.w   d0,(a1)+
291:      lea.l    2(a1),a1
292:      move.w   d0,(a1)+
293:      lea.l    2(a1),a1
294:      move.w   d0,(a1)+
295:      lea.l    2(a1),a1
296:      rts
297:
298: #-----
299: # 描画アドレス決定
300: #-----
301:
302: page_set:
303:      move.l   #000000,a0
304:      moveq.l  #00,d0
305:      move.l   #star_work0,star_point_addr
306:
307:      eor.w    #1,g_flag
308:      bne     obp2
309:
310:      move.l   #00200,a0
311:      move.w   #256,d0
312:      move.l   #star_work1,star_point_addr
313:      obp2:      move.w   d0,gra_scroll    #スクロール座標をワークに格納
314:            jar      SETWPLN          #描画アドレスの設定
315:            rts
316:
317:
318: # スーパーバイザモードへ
319:
320: SUPER:
321:      move.l   a1,-(sp)
322:      suba.l   a1,a1
323:      IOCS     B_SUPER
324:      move.l   d0,ssphuf
325:      move.l   (sp)+,a1
326:      rts
327:
328: # ユーザーモードへ
329:
330: USER:
331:      move.l   a1,-(sp)
332:      movea.l  ssphuf,a1
333:      IOCS     B_SUPER
334:      move.l   (sp)+,a1
335:      rts
336:
337: ssphuf:      dc.l    0
338:
339:
340: #-----
341: # パラメータワーク
342: #-----
343:
344: star_para:
345:      dc.w    0      #DX
346:      dc.w    0      #DY
347:      dc.w    0      #DZ
348:
349:      dc.w    0      #FITCH
350:      dc.w    0      #HEAD
351:      dc.w    0      #BANK
352:
353:      dc.w    0      #SHX
354:      dc.w    0      #SHY
355:
356: #-----
357: # 座標データ
358: #-----
359:
360: star_map_data:
361:      dc.w    star_cnt*2
362:      ds.w    3*star_cnt*2
363:
364: star_work0:
365:      ds.w    star_cnt*8*2
366:
367: star_work1:
368:      ds.w    star_cnt*8*2
369:
370: # ワークエリア
371:
372: star_point_addr:
373:      dc.l    star_work0      # 星の描画バッファアドレス
374:
375: star_back:
376:      ds.w    8      #ダミーのM.N.バッファ
377:      dc.w    $ffff
378:
379: g_flag:
380:      dc.w    00      # 描画ページ切り替えフラグ
381:
382: gra_scroll:
383:      dc.w    0000      # グラフィックススクロール座標
384:
385: x_id:
386:      dc.w    0000      # キー入力による移動量
387:
388: y_id:
389:      dc.w    0000
390:
391: speed:
392:      dc.w    0001      # 星の移動スピード
393:
394:
395: star_grad_data:
396:      dc.w    $0159,$11DB,$221F,$2AA1,$3B23,$4B65,$5BE
397:
398:
399:      dc.w    $74AD,$852F,$95B3,$A5F5,$B677,$C6EF,$CF3
400:
401:
402:      .even

```



## SIDE A

# 視点を制し空間を把握せよ

Tan Akihiko 丹 明彦

ディスプレイの中に広がる仮想空間

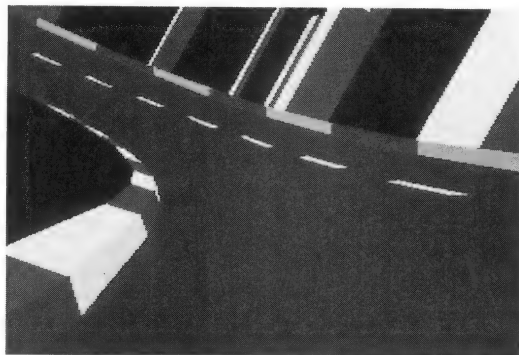
その空間を制御するために知っておかねばならないこと、それが座標系である  
その座標系を制した者だけが、空間を手中に収めることができるのだ

### 「リッジレーサー」に寄せて

近ごろ最も印象的なアーケードゲームは、ナムコの「リッジレーサー」。この号が出るころには世間に出てくるのだろうか。

一時期セガの「ヴァーチャ・レーシング」に主役の座を奪われた感のあるナムコが、ついに逆襲を開始したという印象を受けた。3次元コンピュータグラフィックスを駆使したドライビングシミュレーションゲームなのだが、そのシステムがなかなかすごい。リアルタイムでスムーズシェーディングやテクスチャマッピングを行い、その機能だけでいえばグラフィックワークステーションをコストパフォーマンスで軽く凌駕する。アーケードゲームならこれくらいはやってもらいたいというレベルのものが、ようやく登場したのである。

本連載のように、X68000/030の限界を究め尽くすべく努力を重ねたところで、アーケードゲームメーカーはそれ以上の仕事をするものであり、しょせんはかなわぬという見方もある。横内氏が危機感をつのらせているようだ。が、私はむしろ喜ばしく受け止めている。理由のひとつは、そもそも個人が持てるシステムではないということ（買おうと思えば買える、というつつこみは勘弁）。個人が持てるシス



今回制作した無限円形サーキット。バンクによる視点の動きを体験できる

テムでできる限りのことをやるというのが私の興味である。理由の2つめは、料金を取る以上、アーケードゲームではデザイン上無理なものもあるということ。シミュレーション性を上げて運転しにくくなくても、運転そのものを楽しむためのひたすら気持ちいいゲームにしても、はたまたセットアップとタイムアタックを楽しむ形式にしても、商売になったものではない。その点、パーソナルコンピュータであれば、とりえず走っても楽しいし、やりこんだ先に見えてくる奥の深さも実現できる。要するに棲み分けができるのだ。

私自身はまだ「リッジレーサー」が動いているところを見たことがない。運転したことももちろんない。ゲームデザインも含め、最終的な評価は実物に触れるまで待つ必要があるだろう。

### 3次元空間と座標系

さて今回は、ドライビングシミュレータやフライトシミュレータを目指す第一ステップとして、座標変換を制することにする。座標変換をきちんとやりはじめると、透視変換などのアルゴリズムまで解説する必要があるのだが、それはやらない。我々はSLASHという強力な道具を手に入れているのだ。SLASHの制御パラメータをどう使うかというスタンスでいく。SLASHを使って空間の中を自由に動きまわればいいのだ。

図1は、適当にでっちあげたフライトシミュレータのイメージである。どっしり構えて動かない地面と、その地面に張りついた山や川や橋や道、それに飛行場。空中をのんきに飛ぶプロペラ機。そしてプレイヤーの乗るジェット戦闘機。左に傾いているのは旋回しているのだろうか。

フライトシミュレータといえば、ジェット戦闘機のコクピットから見た光景である。それが図2だ。どっしり構えて動かないはずの地面が大きく右に傾



き、地面に張りついた物体も道連れになっている。プロペラ機も変な方向を目指して飛んでいる。そして今度は、コクピットが画面の真ん中に陣取って動かない。

さあ、たった2枚の図だが、これだけでも座標変換に関するいくつかの示唆を含んでいる。今回からの座標変換の解説はほんのちょっとだけ抽象的かもしれないので、もし万一混乱することがあれば、この図に立ち返って、我々がなにをしようとしているのかを思い出していただきたいと思う。我々の目標は3次元のリアルタイムシミュレータである。ベクトル操作のお勉強ではない。目的意識があれば、数学など怖くはない。

図1や図2のように、ただフライトシミュレータのイメージがあるというだけでは話が先に進まないで、空間を把握するとかかりとして、座標系を導入する。図3をご覧ください。

SLASHで扱う物体はすべて座標系を持っている。それが物体座標系である。モデラで3面図を用いてポリゴンの頂点の座標を定義する。これは物体座標系である。

図3で、飛行機に3つの矢印がついているのがわ

かる。これが物体座標系の座標軸である。物体座標系は、各物体に1つずつ存在し、その物体の運動につれていろいろな場所へ動き、いろいろな方向を向くのである。

地上の物体に座標系がついていないのは、これらが地上に固定されている物体だからだ。もっというなら、地面はそれに張りついた物体を含む、ひとつの巨大な物体なのだ。

その地上の座標系をワールド座標系と呼ぶ。ワールド座標系は、それ自体は動かずに、ほかの物体の運動を記述するなどの用途に使われる。

コクピットの座標系は、実際に画面に表示される座標系である。すなわちSLASHの座標系である。すべての物体は、この座標系に持ち込んで初めて表示できる。

SLASH座標系は右をX軸正方向、下をY軸正方向、奥をZ軸正方向と規定している(図4)。これは、X,Y座標が画面のX,Yに一致し、なおかつ右手系にした結果である。ワールド座標系や物体座標系もこれに準じている。この軸の取り方は必ずしも世間の標準というわけではなく、たとえばグラフィックワークステーションにおいては画面左下が原点(つ

図1 フライトシミュレータで計算機の中に実現する世界



図3 ワールド座標系と物体座標系

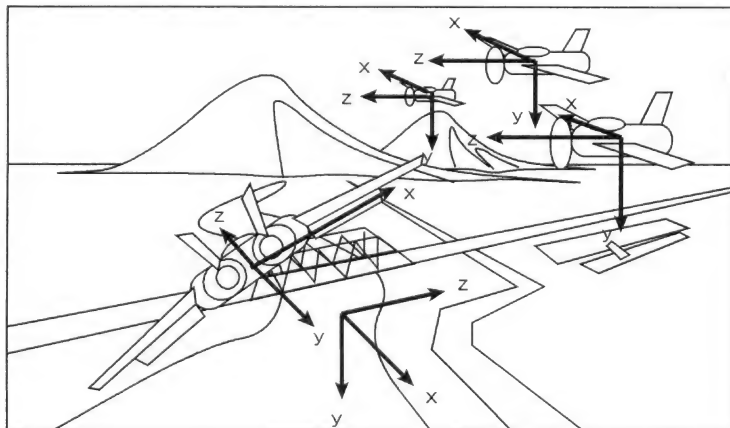


図2 コクピットからのビュー

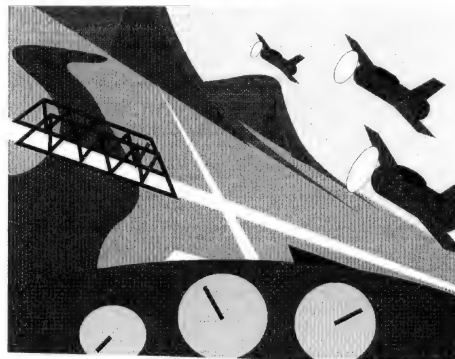
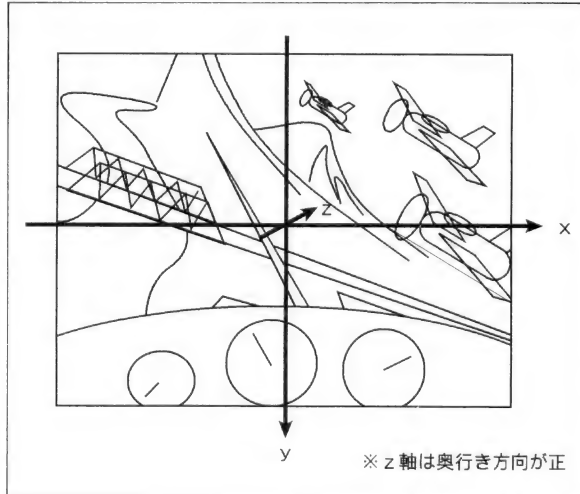


図4 SLASHの座標系





# ハードコア3Dエクスタシー(第2回)

図5 今回の舞台 (円形サーキット)

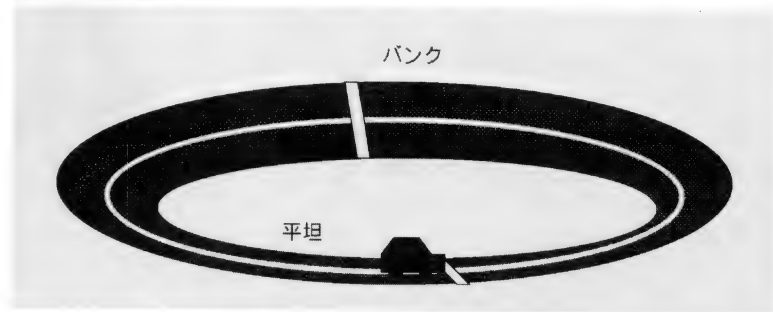
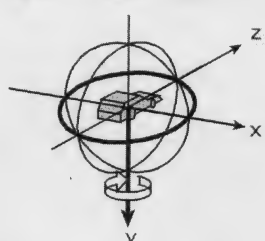


図6 SLASHの座標変換

3次元のベクトル  $\vec{v} = (x_v, y_v, z_v)$  を  $y$  軸まわりに  $\theta$  ラジアン回転させる

(A)  $y$  軸まわりの回転 (head)

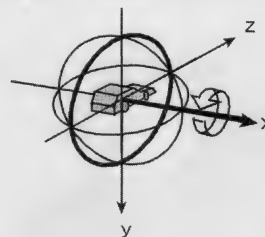


$$R_y(\theta) = \begin{pmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{pmatrix}$$

$$\begin{pmatrix} x_{new} \\ y_{new} \\ z_{new} \end{pmatrix} = R_y(\theta) \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}$$

3次元のベクトル  $\vec{v} = (x_v, y_v, z_v)$  を  $x$  軸まわりに  $\theta$  ラジアン回転させる

(B)  $x$  軸まわりの回転 (pitch)

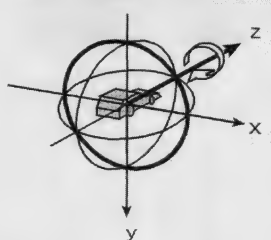


$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{pmatrix}$$

$$\begin{pmatrix} x_{new} \\ y_{new} \\ z_{new} \end{pmatrix} = R_x(\theta) \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}$$

3次元のベクトル  $\vec{v} = (x_v, y_v, z_v)$  を  $y$  軸まわりに  $\theta$  ラジアン回転させる

(C)  $z$  軸まわりの回転 (bank)

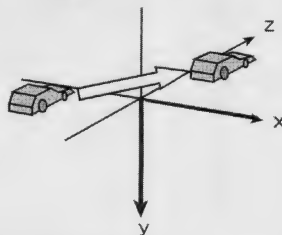


$$R_z(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} x_{new} \\ y_{new} \\ z_{new} \end{pmatrix} = R_z(\theta) \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}$$

3次元のベクトル  $\vec{v} = (x_v, y_v, z_v)$  を  $\vec{d} = (x_d, y_d, z_d)$  平行移動する

(D) 平行移動



$$\begin{pmatrix} x_{new} \\ y_{new} \\ z_{new} \end{pmatrix} = \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix} + \begin{pmatrix} x_d \\ y_d \\ z_d \end{pmatrix}$$

まりYの符号が逆) だったり、たとえば航空力学においてはZ軸が鉛直下方向だったりする。だが今回の解説においては最初いった通りの座標系で通す。対応さえきちんと取れていれば、なんの問題もないはずである。

今回は、ワールド座標系とSLASH座標系の関係を探る。つまり、登場するのは地面だけである (コクピットの計器類も出てこない)。物体座標系とワールド座標系とSLASH座標系の関係は次回のお楽しみである。

## 永久無限地獄コース

今回の舞台は永久無限地獄コースこと円形サーキットである (図5)。この道をひたすら走る自動車から、座標変換の神髄をつかみ取ろう。

コースの一方は平坦で、もう一方はバンクしている。図では省略したが、道の内側には赤白の縁石があり、道の外側には壁がある。さらにコース脇にはビルも建っているのだが、これらはみんな飾り。動きをわかりやすくするためのものである。

この自動車の運動は、円形サーキットをひたすらぐるぐる周回する、ただそれだけ。単純な運動ながら、姿勢が刻々と変化するので、今回の座標系の学習教材としては悪くないだろう。実際、たったこれだけの動きでも、座標変換に関する知識を総動員する必要があるのである。

## SLASHのおさらい

一気に話のテンションを上げる。この手の話をする際に避けて通れない三角関数がいよいよ登場する。

SLASHの座標変換は、基本的に、

- ・ X, Y, Z 軸まわりの回転
- ・ 平行移動

の2つに分けられる。それが図6である。

回転はさらに3つに分けたほうが理解しやすい。それぞれ、ナントカ軸まわりの回転という無味乾燥な名前のほかに、意味のある名前がついている。その意味をわかりやすくするために回転中心に車を置いてみた。方向を変えるのがhead、アップダウンを変えるのがpitch、そして傾きを変えるのがbankである。それぞれ、SLASHの座標軸に合わせてY, X, Z 軸まわりの回転になっている。SLASHの座標変換は、物体座標系のオブジェクトに回転を行ったあとに、平行移動を行うことによってSLASH座標系に持ってくるというプロセスで行われる。

座標変換は、ポリゴンの頂点の位置ベクトルに対して回転や平行移動を行うことで実現されている。たとえば、Y軸まわりに回転させようとするなら、



車を構成するポリゴンの各頂点の位置ベクトルに行列 $R_y(\theta)$ を乗ずるとよい。

回転はbank-pitch-headの順に行われる。もともとの位置ベクトルから、飛行機や車などのオブジェクトを回した場合にそれらしい動きになる、という理由からそうなっている。

ここで重要なことをひとつ。一般に、回転行列は、掛ける順番を変えると結果が変わってしまう。試しに、図6の車をY軸まわりに90度回転させてからX軸まわりに90度回転させると、車は横倒しになる。が、先にX軸まわりに90度回転させてからY軸まわりに90度回転させると、車は直立するのである。これが回転の性質の実に奥深いというか、嫌らしいところなのである。そしてSLASHにおいては回転の順序は変えられない。細心の注意を払ってhead, pitch, bank角を指定しないと正しい姿勢制御はできないということだ。

## どう座標変換を行うか

以上の予備知識をもとに、例の円形サーキットを走る車から見た光景を得るにはどうしたらよいか考える。

図7をご覧ください。今回のモデルにおいては、カメラ（視点と視線）は車に固定されているので、車から見た光景を実現するためには、車の姿勢からSLASHの座標変換パラメータを求め、サーキットのオブジェクトを座標変換する必要がある。

まずありがちな考えとして、道の傾きから車のhead, pitch, bank角は比較的容易にわかると考えられるので、それを使ってサーキットのオブジェクトを回転するというものがある。結論からいえば、これはうまくいかない。まず、そうして求めたhead, pitch, bank角は車の姿勢であり、サーキットの変換パラメータとはならない。

視点が車に固定ということ、サーキットの座標変換は車の座標変換の逆変換になっていないと知らないのである。

冒頭（図1、図2）で飛行機が左に傾くと風景は右に傾いて見える、といったのは、ここのための伏線だったのだ。

逆変換なのだから、求めたhead, pitch, bank角にそれぞれ-1をかければいいのでは？ というのも素人考えといえる。まず、一般に、回転行列の角度の符号を反転しても逆変換にはならない。そうしたうえで、回転行列を掛ける順番を逆にしなくてはならない。

たとえば、X軸まわりに90度回転したあとにY軸まわりに90度回転した車を見てみよう。車は直立して横を向いているはずだ。これをさらにX軸まわり

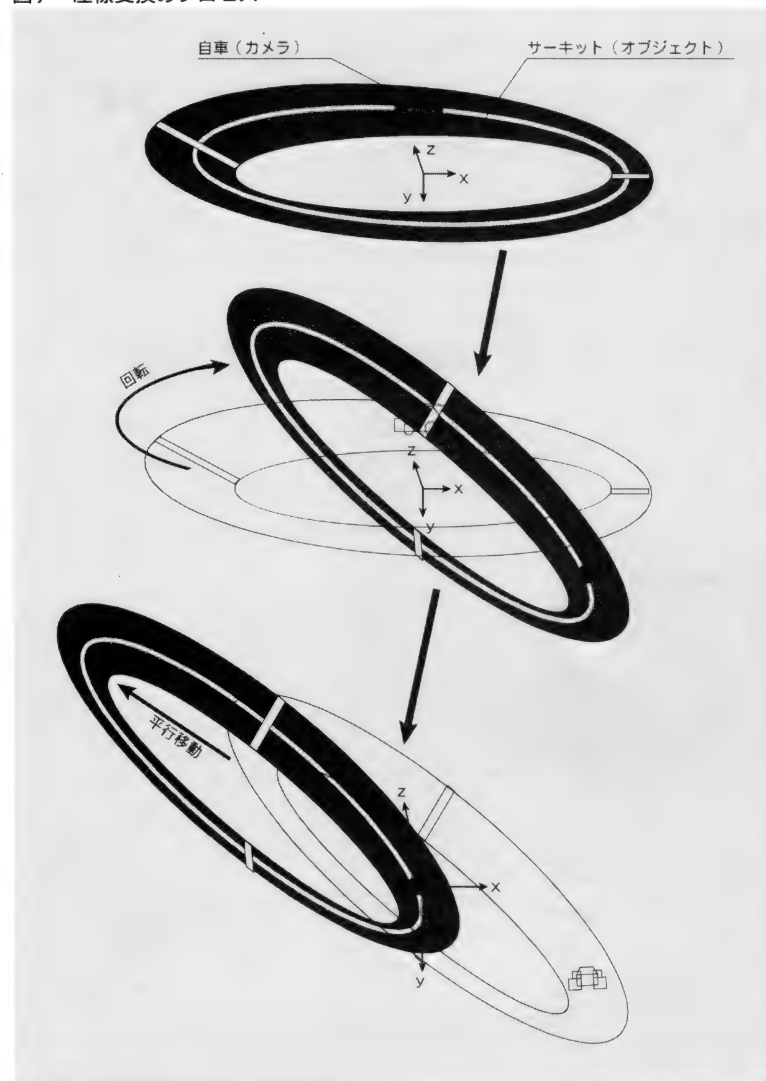
に-90度回転したあとにY軸まわりに-90度回転させてみる。どうなっただろうか？ 車は横倒しになっているはずだ。正しくは、Y軸まわりに-90度回転したあとにX軸まわりに-90度回転させる。もとに戻るはずだ。

回転はかくのごとく奥が深くて嫌らしいのである。SLASHの回転の順番が決められないという特徴は、ここにきて重くのしかかることになる。

## 基底座標系の導入

そこで基底座標系という概念を導入する（図8）。物体座標系とはほぼ同じである。違うところは2つあって、ひとつはワールド座標の物体の位置に配置することを前提にしているということ。もうひとつは、各座標軸を表すベクトル（これを基底ベクトルと呼び、 $\alpha, \beta, \gamma$ がそれぞれX, Y, Z軸に対応する）が単

図7 座標変換のプロセス





# ハードコア3Dエクスタシー(第2回)

位ベクトルであるということである。

基本戦略は次のようになる。

- 1) コース上の車の物体座標軸を基底座標として求める
- 2) 基底座標系を回転してその基底ベクトルをSLASH座標系の3軸に合わせる
- 3) このとき求めた回転をサーキットオブジェクトに適用すると、望みどおりの回転が得られる

## オイラー角の算出

軸を合わせる手順については図9をご覧ください。ことにしよう。簡単にいえば、

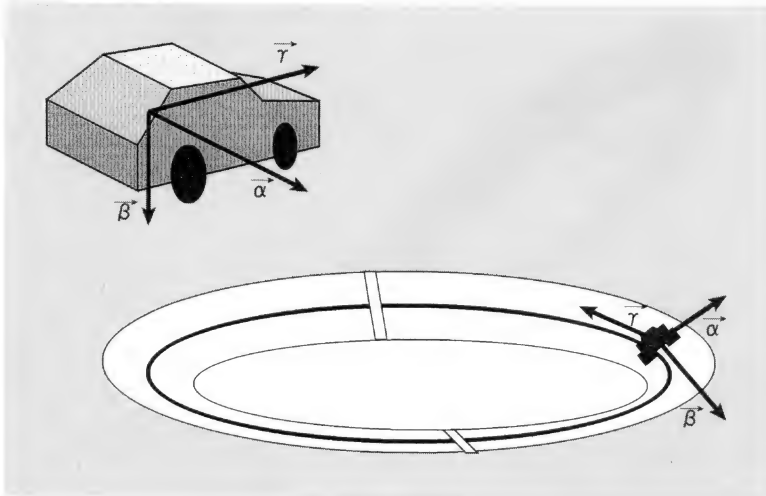
- 1) bank角を求めて $\beta$ 軸をYZ平面に乗せる
- 2) pitch角を求めて $\beta$ 軸をY軸に一致させる (この時点で $\alpha$ 軸と $\gamma$ 軸はXY平面に乗る)
- 3) head角を求めて $\alpha$ 軸をX軸に、 $\gamma$ 軸をZ軸に一致させる
- 4) 求めた角度から回転行列を作り、車の位置ベクトルに掛け、その結果を平行移動量として用いる
- 5) 3つの回転角と平行移動量をSLASH座標変換パラメータとして用いる

ということになる。気をつけることといえば、角度の算出には逆正接関数 $\arctan$  (Cライブラリでは $\text{atan}$ ) を用いているが、 $\arctan$ は周期 $\pi$  ( $180^\circ$ ) なので、場合分けして正しい角度を求める関数 $\text{angle}()$ を作ったことくらいである。

## 基底座標系についての余談

なぜ基底座標系というかという、物体座標系の値を基底座標に持ち込むと、そのままワールド座標内の物体の座標になるからである。また (同じことをいっているのだが)、基底座標系の3軸のベクト

図8 基底座標系



ルの要素を並べて $3 \times 3$ 行列にすると、物体座標系からワールド座標系への回転行列になるのである。

余談だが、回転行列の逆行列はそれを転置したものである (転置とは、正方行列の要素を対角要素を境にしてひっくりかえすこと)。これを利用すれば、今回の処理も、面倒なオイラー角の計算をすることなく、基底座標から求めた行列を転置するだけで可能なのである。回転行列のこの性質は、場合によっては (今回もそのケースに入ることは入る) とても役に立つので、本当は余談にしてはいけないのだが、現行のSLASHでは、回転を角度パラメータでしか指定できないので余談にしてある。ちなみに数カ月後にリリース予定のSLASHの次期バージョンでは、回転を角度だけでなく行列で直接指定することも可能になる予定である (この副作用として、回転の精度がサインテーブルの精度に制限されるということもなくなる)。そうなれば、今回やったことはまったく無意味ということになるが、まあそれはそれ、物事を筋道立てて解く今回のアプローチは、ほかの場面でも必ず必要になることであろう。

## 今月のプログラム

### ●eulerlib.c, eulerlib.h

オイラー角を求める関数 $\text{euler}()$ などが入っている。

### ●runtest.c

テストプログラム。円形サーキットを作り、その中を走る。eulerlibを利用する。

### ●runtest.xの使い方

まずリストを打ち込んで適当な場所に置く。適当にMakefileを書いてコンパイルする。

runtestとタイプすれば起動する。

マウスの前後がアクセル、左右がハンドルのようなものである。ドライビングシミュレーションではないからあまり期待しないこと。

マウスの左右ボタンで視点の高さが変わる。あまり低くすると、視点が道路の下に潜るから注意。

F1キーで、 $\text{euler}()$  関数の求めた角度を表示するモードに入る。もう一度F1キーを押すと元のモードに戻る。角度の表示が目まぐるしく変わることがわかりになるであろう。

## 終わりに

runtest.xを作ってみて思ったのは、これだけいじめてもまだまだ動くSLASHが偉大だということ。ポリゴン数は見かけより多い。簡単に計算してみたが、頂点数は700点を超えていた。プログラムによるコースデータの自動生成というのは楽で、無節操に



つけ足していったためこんなことになってしまったのだ。ともあれ、このクラスのデータになると座標変換の処理時間が馬鹿にならなくなってくる。特に10MHz機だと、1フレームあたり0.1秒のオーダーにのぼってくるのだ。

先月号が発売されて間もなく、読者から「SLASH改悪」と題した一通の投稿が届いた。むろん内容は改悪などではない。驚くべきことに、座標変換をさらに高速化するものである。回転行列には、三角関数の積が出てくるが、いくつかの公式を用いればこれを和に直せることを利用した手法である。MC68000においては、いうまでもなく積より和がはるかに高速であり、1頂点あたり数百クロックが稼げるといふ。まさに一本とられた気分である。と同時に、こういう小気味いいレスポンスを素早く返してくる読者の存在に深い感動を禁じえない。世の中、すごいやつはいっぱいいるんだなあというところである。ほんの数日のうちにSLASHのソースを読み、改良点を発見するとは！

そして、中学や高校の数学で習う公式は役に立たないように見えて、実はとんでもない場面で力を発揮することは肝に銘じておく必要がある。また、そうした公式は、正確な表式はともかく、どこか心の片隅にでも置いておかないと、作っているプログラムを高速化できる可能性にさえずらないのである。結局ものをいうのは教養なのである、と教訓めいた結論が出たところで今回の話を終わることにする。驚きと感動と教訓を与えてくださった坪井さんに心から感謝申し上げたい。

## 次回予告

一応2つの候補がある。

1) 座標変換の次の段階として、物体座標とワールド座標とSLASH座標の関係をまとめる。コースの中をはかの車が走り回り、コクピットもつくことになるだろう。

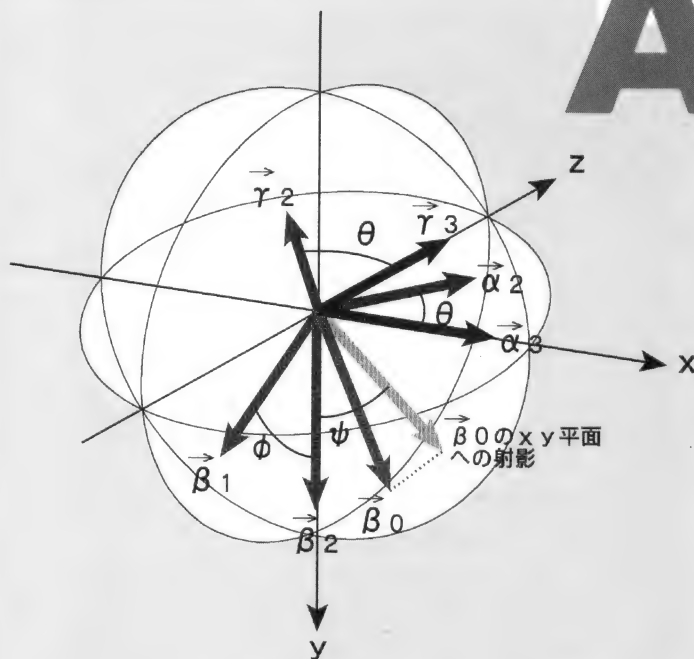
2) 今回は自動車の（つまり視点の）移動をプログラムで自動生成したため、動きが不自由である。これをどこでも走り回れるようにする。

どちらが先になるかはわからないが、1)からいくのが筋のようではある。それでは、また来月。

## ■リスト1 eulerlib.h

```
1: /*
2:  eulerlib.h
3:  - slashlibの補助関数(オイラー角)
4:  Sep. 1993      丹 明彦(Oh!X)
5:  */
6:
7: #ifndef __EULERLIB_H__
8: #define __EULERLIB_H__
9:
10: #include <math.h>
11:
12: #ifndef M_PI /* π: XCのmath.hではシンボル名PIで定義 */
13: #define M_PI PI
14: #endif
```

図9 関数euler()の動作



	初期設定	Phase 2 x軸まわりの回転 (pitch)
位置ベクトル	$\vec{p} = (x_p, y_p, z_p)$	$\phi = -\arctan(z_{\beta_1}/y_{\beta_1})$
基底ベクトル	$\vec{\alpha}_0 = (x_{\alpha_0}, y_{\alpha_0}, z_{\alpha_0})$	$\vec{\alpha}_2 = R_x(\phi)\vec{\alpha}_1$
	$\vec{\beta}_0 = (x_{\beta_0}, y_{\beta_0}, z_{\beta_0})$	$\vec{\beta}_2 = R_x(\phi)\vec{\beta}_1$
	$\vec{\gamma}_0 = (x_{\gamma_0}, y_{\gamma_0}, z_{\gamma_0})$	$\vec{\gamma}_2 = R_x(\phi)\vec{\gamma}_1$
Phase 1 z軸まわりの回転 (bank)		Phase 3 y軸まわりの回転 (head)
$\psi = \arctan(x_{\beta_0}/y_{\beta_0})$		$\theta = -\arctan(z_{\alpha_2}/x_{\alpha_2})$
$\vec{\alpha}_1 = R_z(\psi)\vec{\alpha}_0$		$\vec{\alpha}_3 = R_y(\theta)\vec{\alpha}_2$
$\vec{\beta}_1 = R_z(\psi)\vec{\beta}_0$		$\vec{\beta}_3 = R_y(\theta)\vec{\beta}_2$
$\vec{\gamma}_1 = R_z(\psi)\vec{\gamma}_0$		$\vec{\gamma}_3 = R_y(\theta)\vec{\gamma}_2$
	Phase 4 平行移動	
	$\vec{p}_{view} = R_y(\theta)R_x(\phi)R_z(\psi)\vec{p}$	

## ■参考文献

Foley, van Dam, Feiner, Hughes: Computer Graphics - principles and practice - Second Edition, Addison-Wesley Publishing Co., 1990

```
15:
16: typedef double VECTOR3[3];
17:
18: #define ITOD(I) (M_PI*2.0*(double)(I)/4096.0)
19: #define DTOI(D) (((int)(4096.0*(D)/(M_PI*2.0))+4096)%4096)
20:
21: double angle(double, double);
22: void rotateZ(VECTOR3*, double);
23: void rotateX(VECTOR3*, double);
24: void rotateY(VECTOR3*, double);
25: void euler(SLPARAMETER*, VECTOR3*, VECTOR3*, VECTOR3*, VECTOR3*);
26:
27: #endif /* __EULERLIB_H__ */
```



# ハードコア3Dエクスタシー(第2回)

## ■リスト2 eulerlib.c

```
1: /*
2:     eulerlib.c
3:     - slashlibの補助関数(オイラー角)
4:     Sep. 1993 丹 明彦(Oh!X)
5: */
6:
7: #include "libV_slashlib.h"
8: #include "eulerlib.h"
9:
10: /* 原点から点(x,y)へのベクトルの
11:    x軸となす角度を求める
12:    0 ≤ 戻り値 < 2π */
13: double angle( x, y )
14: {
15:     double t;
16:     if ( x > 0 ) {
17:         t = atan( y / x );
18:         if ( t < 0.0 ) t += (M_PI*2.0);
19:         return t;
20:     } else if ( x < 0.0 ) {
21:         t = atan( y / x ) + M_PI;
22:         return t;
23:     } else {
24:         /* x=0 */
25:         if ( y > 0.0 ) {
26:             return ( M_PI/2.0 );
27:         } else if ( y < 0.0 ) {
28:             return ( 3.0*M_PI/2.0 );
29:         } else {
30:             /* x=y=0 */
31:             /* 角度がないことによる利用上問題なし */
32:             return ( 0.0 );
33:         }
34:     }
35: }
36: /* z軸周りの回転(bank相当) */
37: void rotateZ( v, t )
38: VECTOR3 *v; /* ベクトル */
39: double t; /* 角度(ラジアン) */
40: {
41:     double v0, v1, v2;
42:     v0 = (*v)[0]; v1 = (*v)[1]; v2 = (*v)[2];
43:     (*v)[0] = v0*cos(t) - v1*sin(t);
44:     (*v)[1] = v0*sin(t) + v1*cos(t);
45:     (*v)[2] = v2;
46:     return;
47: }
48: /* x軸周りの回転(pitch相当) */
49: void rotateX( v, t )
50: VECTOR3 *v; /* ベクトル */
51: double t; /* 角度(ラジアン) */
52: {
53:     double v0, v1, v2;
54:     v0 = (*v)[0]; v1 = (*v)[1]; v2 = (*v)[2];
55:     (*v)[0] = v0;
56: }
```

```
57:     (*v)[1] = v1*cos(t) - v2*sin(t);
58:     (*v)[2] = v1*sin(t) + v2*cos(t);
59:     return;
60: }
61:
62: /* y軸周りの回転(head相当) */
63: void rotateY( v, t )
64: VECTOR3 *v; /* ベクトル */
65: double t; /* 角度(ラジアン) */
66: {
67:     double v0, v1, v2;
68:     v0 = (*v)[0]; v1 = (*v)[1]; v2 = (*v)[2];
69:     (*v)[0] = v0*cos(t) + v2*sin(t);
70:     (*v)[1] = v1;
71:     (*v)[2] = -v0*sin(t) + v2*cos(t);
72:     return;
73: }
74:
75: /* ワールド座標系の位置と基底ベクトルから
76:    オイラー角と平行移動量を求めて
77:    SLPARAMETER型変数に格納する */
78: void euler( p, v, vx, vy, vz )
79: SLPARAMETER *p;
80: VECTOR3 *v, *vx, *vy, *vz;
81: {
82:     double dbank, dpitch, dhead;
83:
84:     /* bank (z軸周り): vyをyz平面に投影する */
85:     dbank = angle( (*vy)[1], (*vy)[0] );
86:     rotateZ( vx, dbank );
87:     rotateZ( vy, dbank );
88:     /* pitch (x軸周り): vxをx軸に一致させる */
89:     dpitch = -angle( (*vx)[1], (*vx)[2] );
90:     rotateX( vx, dpitch );
91:     rotateX( vy, dpitch );
92:     rotateX( vz, dpitch );
93:     /* head (y軸周り): vxとvzをx軸に一致させる */
94:     dhead = angle( (*vx)[0], (*vx)[2] );
95:     /* この後のrotate
96:     eYは必要 */
97:     rotateY( v, dhead );
98:     /* オイラー角に従って回転させ平行移動量を求める */
99:     rotateZ( v, dbank ); /* bank (z軸周り) */
100:    rotateX( v, dpitch ); /* pitch (x軸周り) */
101:    rotateY( v, dhead ); /* head (y軸周り) */
102:
103:    /* 座標変換パラメータに代入する */
104:    p->bank = DTOI( dbank );
105:    p->pitch = DTOI( dpitch );
106:    p->head = DTOI( dhead );
107:    p->x = -(int)(*v)[0];
108:    p->y = -(int)(*v)[1];
109:    p->z = -(int)(*v)[2];
110:
111:    return;
112: }
```

## ■リスト3 runtest.c

```
1: /*
2:     runtest.c
3:     - slashlibのテストプログラム
4:     Sep. 1993 丹 明彦(Oh!X)
5: */
6:
7: #define _IOCS_INLINE_
8: #include <iocslib.h>
9: #define _DOS_INLINE_
10: #include <doslib.h>
11: #include <stdio.h>
12: #include <stdlib.h>
13:
14: #include "libV_slashlib.h"
15: #include "colorV_tplib.h"
16: #include "eulerlib.h"
17:
18: SLPOLYGONLIST *road_polygonlist;
19: SLPOINTLIST *road_pointlist;
20:
21: SLPOLYGONLIST *build_polygonlist;
22: SLPOINTLIST *build_pointlist;
23:
24: SLTRANSWORK *work;
25: SIMINMAX *minmax1, *minmax2, *minmax3;
26: SLPARAMETER *parameter;
27:
28: #define ROAD_DIV 32
29: #define ROAD_R1 500
30: #define ROAD_R2 756
31: #define ROAD_HEIGHT 128
32:
33: /* コースは全周を巡って一定の傾き */
34: /* 傾き RH(t) ROAD_HEIGHT */
35: /* コースの傾きが変化 */
36: RH(t) ((int)((double)ROAD_HEIGHT*sin(t)-M_P
37: 1/2.0)*(double)ROAD_HEIGHT/2);
38:
39: #define LINE_DIV 36
40: #define LINE_WIDTH 6
41: #define STONE_DIV 64
42: #define STONE_WIDTH 8
43: #define WALL_DIV 36
44: #define WALL_HEIGHT 16
45: #define BUILD_N 32
46: #define BUILD_X 2048
47: #define BUILD_Y 2048
48: #define BUILD_W 64
49: #define BUILD_W_OFF 16
50: #define BUILD_H 512
51: #define BUILD_H_OFF 256
52:
```

```
53: void setup_road()
54: {
55:     int i;
56:     int r1, r01, h11, h01;
57:     int r2, r02, h12, h02;
58:     SLPALET *c;
59:     double theta1, theta2, rbank1, rbank2;
60:     int rh1, rh2;
61:
62:     road_polygonlist->n = 0;
63:     road_pointlist->n = 0;
64:     for ( i = 0; i < ROAD_DIV; i++ ) {
65:         theta1 = (double)(i)*M_PI*2/ROAD_DIV;
66:         theta2 = (double)((i+1)*M_PI*2/ROAD_DIV);
67:         rh1 = RH(theta1);
68:         rh2 = RH(theta2);
69:         if ( i == 0 ) c = &std_darkgreen;
70:         else if ( i == ROAD_DIV/2 ) c = &orange;
71:         else c = &std_darkgray;
72:         addtetragon( road_polygonlist, road_pointlist,
73:             (int)(ROAD_R2*cos(theta1)), -rh1, (int)(ROAD_R2*sin(theta1)),
74:             (int)(ROAD_R1*cos(theta1)), 0, (int)(ROAD_R1*sin(theta1)),
75:             (int)(ROAD_R1*cos(theta2)), 0, (int)(ROAD_R1*sin(theta2)),
76:             (int)(ROAD_R2*cos(theta2)), -rh2, (int)(ROAD_R2*sin(theta2)), c );
77:     }
78:     for ( i = 0; i < LINE_DIV; i++ ) {
79:         theta1 = (double)(i)*M_PI*2/LINE_DIV;
80:         theta2 = theta1 + M_PI/LINE_DIV;
81:         rh1 = RH(theta1);
82:         rh2 = RH(theta2);
83:         rbank1 = atan( (double)rh1/(double)(ROAD_R2 - ROAD_R1) );
84:         rbank2 = atan( (double)rh2/(double)(ROAD_R2 - ROAD_R1) );
85:         r1 = (ROAD_R1 + ROAD_R2)/2 - LINE_WIDTH*cos(rbank1)/2;
86:         r2 = (ROAD_R1 + ROAD_R2)/2 - LINE_WIDTH*cos(rbank2)/2;
87:         h11 = -rh1/2 + LINE_WIDTH*sin(rbank1)/2;
88:         h12 = -rh2/2 + LINE_WIDTH*sin(rbank2)/2;
89:         r01 = (ROAD_R1 + ROAD_R2)/2 + LINE_WIDTH*cos(rbank1)/2;
90:         r02 = (ROAD_R1 + ROAD_R2)/2 + LINE_WIDTH*cos(rbank2)/2;
91:         h01 = -rh1/2 - LINE_WIDTH*sin(rbank1)/2;
92:         h02 = -rh2/2 - LINE_WIDTH*sin(rbank2)/2;
93:         addtetragon( road_polygonlist, road_pointlist,
94:             (int)(r01*cos(theta1)), h01, (int)(r01*sin(theta1)),
95:             (int)(r11*cos(theta1)), h11, (int)(r11*sin(theta1)),
96:             (int)(r12*cos(theta2)), h12, (int)(r12*sin(theta2)),
97:             (int)(r02*cos(theta2)), h02, (int)(r02*sin(theta2)), &std_
98:             d_white );
99:     }
100:     r01 = r02 = ROAD_R1;
101:     r11 = r12 = ROAD_R1 - STONE_HEIGHT;
102: }
```



```

101: r11 = r12 = ROAD_R1 - STONE_HEIGHT - STONE_WIDTH;
102: h11 = h12 = -STONE_HEIGHT;
103: h11 = h12 = -STONE_HEIGHT;
104: h01 = h02 = 0;
105: for ( i = 0; i < STONE_DIV; i++ ) {
106:     if ( i%2 == 0 ) c = &std_white;
107:     else c = &std_red;
108:     theta1 = (double)(i)*M_PI*2/STONE_DIV;
109:     theta2 = (double)((i+1)*M_PI*2/STONE_DIV;
110:     addtetragon( road_polygonlist, road_pointlist,
111:         (int)(r01*cos(theta1)), h01, (int)(r01*sin(theta1)),
112:         (int)(r11*cos(theta1)), h11, (int)(r11*sin(theta1)),
113:         (int)(r21*cos(theta1)), h21, (int)(r21*sin(theta1)),
114:         (int)(r02*cos(theta2)), h02, (int)(r02*sin(theta2)),c);
115:     addtetragon( road_polygonlist, road_pointlist,
116:         (int)(r11*cos(theta1)), h11, (int)(r11*sin(theta1)),
117:         (int)(r21*cos(theta1)), h21, (int)(r21*sin(theta1)),
118:         (int)(r12*cos(theta2)), h12, (int)(r12*sin(theta2)),c);
119:     (int)(r22*cos(theta2)), h22, (int)(r22*sin(theta2)),c);
120: }
121: for ( i = 0; i < WALL_DIV; i++ ) {
122:     if ( i%2 == 0 ) c = &std_yellow;
123:     else c = &std_blue;
124:     theta1 = (double)(i)*M_PI*2/WALL_DIV;
125:     theta2 = (double)((i+1)*M_PI*2/WALL_DIV;
126:     rh1 = RH(theta1);
127:     rh2 = RH(theta2);
128:     rbank1 = atan( (double)rh1/(double)(ROAD_R2 - ROAD_R1) );
129:     rbank2 = atan( (double)rh2/(double)(ROAD_R2 - ROAD_R1) );
130:     r11 = (ROAD_R1 + ROAD_R2)/2 - LINE_WIDTH*cos(rbank1)/2;
131:     r12 = (ROAD_R1 + ROAD_R2)/2 - LINE_WIDTH*cos(rbank2)/2;
132:     h11 = -rh1/2 + LINE_WIDTH*sin(rbank1)/2;
133:     h12 = -rh2/2 + LINE_WIDTH*sin(rbank2)/2;
134:     r01 = (ROAD_R1 + ROAD_R2)/2 + LINE_WIDTH*cos(rbank1)/2;
135:     r02 = (ROAD_R1 + ROAD_R2)/2 + LINE_WIDTH*cos(rbank2)/2;
136:     h01 = -rh1/2 - LINE_WIDTH*sin(rbank1)/2;
137:     h02 = -rh2/2 - LINE_WIDTH*sin(rbank2)/2;
138:     r11 = ROAD_R2;
139:     r12 = ROAD_R2;
140:     r01 = ROAD_R2 - WALL_HEIGHT*sin(rbank1);
141:     r02 = ROAD_R2 - WALL_HEIGHT*sin(rbank2);
142:     h11 = -rh1;
143:     h12 = -rh2;
144:     h01 = -rh1 - WALL_HEIGHT*cos(rbank1);
145:     h02 = -rh2 - WALL_HEIGHT*cos(rbank2);
146:     addtetragon( road_polygonlist, road_pointlist,
147:         (int)(r01*cos(theta1)), h01, (int)(r01*sin(theta1)),
148:         (int)(r11*cos(theta1)), h11, (int)(r11*sin(theta1)),
149:         (int)(r12*cos(theta2)), h12, (int)(r12*sin(theta2)),c);
150:     (int)(r02*cos(theta2)), h02, (int)(r02*sin(theta2)),c);
151: }
152: return;
153: }
154: void setup_build()
155: {
156:     int i, x, y, w, h;
157:     int i, sp, time = 0, run = 0;
158:     int mscur, x, y, msdt, lb, rb;
159:     int run_h = 29, rh;
160:     int debug = 0;
161:     double theta, rbank;
162:     VECTOR3 v, vx, vy, vz;
163:     road_polygonlist = malloc( sizeof(SLPOLYGONLIST)*sizeof(SLPOINT) );
164:     (ROAD_DIV + LINE_DIV + STONE_DIV*2 + WALL_DIV) );
165:     road_pointlist = malloc( sizeof(SLPOLYGONLIST)*sizeof(SLPOINT) );
166:     (ROAD_DIV*2 + LINE_DIV*4 + STONE_DIV*3 + WALL_DIV*2) );
167:     setup_road();
168:     AddNorm( road_polygonlist, road_pointlist );
169:     build_polygonlist = malloc( sizeof(SLPOLYGONLIST)*sizeof(SLPOINT) );
170:     BUILD_N*6;
171:     build_pointlist = malloc( sizeof(SLPOLYGONLIST)*sizeof(SLPOINT) );
172:     BUILD_N*8;
173:     setup_build();
174:     AddNorm( build_polygonlist, build_pointlist );
175:     /* number of points */
176:     work = malloc( sizeof(SLTRANSWORK)*1000 );
177:     /* number of objects */
178:     minmax1 = malloc( sizeof(SLMINMAX)*(2+1) );
179:     /* number of objects + 1 */
180:     minmax2 = malloc( sizeof(SLMINMAX)*(2+1) );
181:     CRTMOD( 14 );
182:     G_CLR_ON();
183:     B_CUROFF();
208: MS_INIT();
209: MS_CUROFF();
210: MS_LIMIT(0,0,255,255);
211: MS_CURST(255,255);
212: MS_CUROFF();
213: SKEY_MOD(0,0,0);
214:
215: SetClearColor( 0 );
216: SetWindowSize( 256, 256 );
217: SetWindowCenter( 128, 128 );
218:
219: parameter.x = 0;
220: parameter.y = 0;
221: parameter.z = 500;
222: parameter.pitch = 0;
223: parameter.head = 0;
224: parameter.bank = 0;
225: parameter.alpha = 16;
226: parameter.beta = 16;
227:
228: sp = SUPER( 0 );
229:
230: for ( ;; ) {
231:     mscur = MS_CURGT();
232:     x = mscur/65536;
233:     y = mscur%65536;
234:     msdt = MS_GETDT();
235:     lb = msdt/256;
236:     rb = msdt%256;
237:     if ( BITSNS(0x00)&2 ) { /* ESCキーで終了 */
238:         while ( BITSNS(0x00)&2 );
239:         break;
240:     }
241:     if ( BITSNS(0x0C)&8 ) { /* F1: デバッグモード */
242:         while ( BITSNS(0x0C)&8 );
243:         printf( "V032n" );
244:         debug = 1 - debug;
245:     }
246:     if ( lb ) run_h++;
247:     if ( rb ) run_h--;
248:
249:     /* 円形のテストコースを巡回する */
250:     /* 車の位置はrunであらわされる(0 ≤ run < 4095) */
251:     run = (255-y);
252:     run = 4095;
253:     theta = ITOD(run);
254:     rh = RH(theta);
255:     rbank = atan( (double)rh/(double)(ROAD_R2 - ROAD_R1) );
256:     /* 車の位置(ワールド座標) */
257:     x = (ROAD_R2-ROAD_R1)*x/256;
258:     v[0] = (ROAD_R1+x - run_h*sin(rbank))*cos(theta);
259:     v[1] = -x*tan(rbank) - run_h*cos(rbank);
260:     v[2] = (ROAD_R1+x - run_h*sin(rbank))*sin(theta);
261:     /* 車の基底ベクトル(x軸相当) */
262:     vx[0] = cos(rbank)*cos(theta);
263:     vx[1] = -sin(rbank);
264:     vx[2] = cos(rbank)*sin(theta);
265:     /* 車の基底ベクトル(y軸相当) */
266:     vy[0] = sin(rbank)*cos(theta);
267:     vy[1] = cos(rbank);
268:     vy[2] = sin(rbank)*sin(theta);
269:     /* 車の基底ベクトル(z軸相当) */
270:     vz[0] = -sin(theta);
271:     vz[1] = 0.0;
272:     vz[2] = cos(theta);
273:     euler( &parameter, &v, &vx, &vy, &vz );
274:     if ( debug == 1 )
275:         printf( "V036 run = %d Vn head = %d Vn pitch = %d Vn bank = %d Vn",
276:             run, parameter.head, parameter.pitch, parameter.bank );
277:
278:     if ( time%2 == 0 ) {
279:         SetWritePlane( (unsigned short *)0xC00000 );
280:         minmax1 = minmax2;
281:         TranslateAll( &parameter, work, build_pointlist, minmax1 );
282:         DisplayPolygonList( build_polygonlist, work, minmax1 );
283:         minmax1 = AdjustMinMax( minmax1 );
284:         TranslateAll( &parameter, work, road_pointlist, minmax1 );
285:         DisplayPolygonList( road_polygonlist, work, minmax1 );
286:         minmax2 = AdjustMinMax( minmax1 );
287:         HOME( 0, 0, 0 );
288:         if ( time > 0 ) {
289:             SetClearPlane( (unsigned short *)0xC00200 );
290:             ClearBox( minmax2 );
291:         }
292:     } else {
293:         SetWritePlane( (unsigned short *)0xC00200 );
294:         minmax2 = minmax1;
295:         TranslateAll( &parameter, work, build_pointlist, minmax2 );
296:         DisplayPolygonList( build_polygonlist, work, minmax2 );
297:         minmax2 = AdjustMinMax( minmax2 );
298:         TranslateAll( &parameter, work, road_pointlist, minmax2 );
299:         DisplayPolygonList( road_polygonlist, work, minmax2 );
300:         minmax1 = AdjustMinMax( minmax2 );
301:         HOME( 0, 256, 0 );
302:         SetClearPlane( (unsigned short *)0xC00000 );
303:         ClearBox( minmax1 );
304:     }
305:     time++;
306: }
307: SUPER( sp );
308: B_CUROFF();
309: CRTMOD( 16 );
310: free( work );
311: free( minmax1 );
312: free( minmax2 );
313: free( road_polygonlist );
314: free( road_pointlist );
315: free( build_polygonlist );
316: free( build_pointlist );
317: KFLUSHIO( 0xFF );
318:
319: return 0;
320: }

```



## SIDE B

# ポリゴン描画のためのエッジ検出法

Yokouchi Takeshi 横内 威至

今月からいよいよポリゴナイザ「SLASH」を研究していく

まずは、ポリゴン描画のためのエッジ検出法を紹介、その利用法を探る

固定概念に捕らわれず、より広く、より深くアルゴリズムを掘り下げていこう

### ポリゴナイザの構造

皆、SLASHシステムを理解できたであろうか。この原稿を書いているのはまだ9月初めだから、読者の声が届いていない。ちょっと気になるところだ。質問なんかが多ければ、来月からサポートしていきたいと思っている。

さて今月はSLASH特集ということだが、俺は一切仕事をしていない。今月は超多忙なので勘弁していただきたい。おそらくスタッフの人たちが、いろいろと遊びながらサンプルを示してくれるだろうから、参考にして技術を積んでいってもらいたい。

ということでこちらは独自に動いているのだが、かなりプレッシャーを感じざるを得ない恐ろしいモノが登場するではないか。ナムコの「リッジレーサー」だ。ついにフルテクスチャマッピング、効果はわからないが、グローシェーディングのエラクリアルなポリゴナイザを搭載してしまった。すでにグラフィックワークステーションクラスを超越しているかもしれない。やはり技術を積み重ねたナムコが、さらに進歩してしまったようである。あんなものを見せつけられるともう生きているのが嫌になるね。まあ、あのクラスが家庭用のコンピュータに載るのは、まだ5年先であろうと甘く計算しつつ、とにかく現在は遅れぬように突っ走るのみである。

では、さっそく今回からは少しポリゴナイザ自体について研究しよう。本来これだけでも膨大な要素を含んでいるのだが、用途はリアルタイム制御ということで、ある程度の範囲に絞って研究しようと思う。もしほかの用途を考えたり、また独自にシステムを勉強したいならば、この1冊をお勧めする。日刊工業新聞社の「実践コンピュータグラフィックス基礎手続きと応用」である。価格は6,500円。昭和62年に発行された本であるが、現在でもこれを超える内容の本を俺は知らない。さすがにやや古めだが、

現在あるグラフィック理論の基礎となるべき内容が網羅されている。3Dシステムを学ぶうえで手元に置いておきたいアイテムであろう。

### ポリゴンについて

物体を平面の集合体として扱い、サーフェイスモデル、つまり表面だけで中身の抜けたモデルを扱うのが現在では一般的である。実際3Dを表現する手段としては現在これを超える方法はないのではなからうか。少し前であるが、DOS/V用のフライトモノ「COMANCHE」を見たとき、俺はもう死のうと思った。ポリゴンモデルでは扱いにくい細かい凹凸(つまりは地形)が見事にリアルに表現されていたのだ。ボクセルスペースと呼ばれる新理論を導入しているらしいが、俺にはなんのことだかわからない。もしかしたら3Dに革命が起こるのでは、と危惧していたのだが、ナムコのアレを見てからはそんな恐怖は消えさった。やはりポリゴンでも加速すればまだまだ奥があるのだ。

くどいがSLASHシステムがベストではない。あくまで俺にできる最高レベルのシステムなだけで、もっとクレイジーな技術者がコーディングすれば、よりイカレたモノができるに違いない。そんな輩が現れるのを期待してはいるし、かといって自分が遅れるのも恐ろしいのだ。

さて、SLASHシステムを見返すと、もうこれ以上の大掛かりな処理はかなり厳しそうである。大掛かりというのは、要するにマッピング及びグロー、フォンシェーディングなんかである。ただし現代のコンピュータの発展を考えるといまから研究しても十分遅れているぐらいである。次世代のハードならばこんなことはソフトウェアで行う必要はなくなりそうだが、技術者である人間は当然知らねばならないことである。俺は技術者でないけど興味深いから研究するのではあるが。現段階ではマッピングまで可



能なハードではないので、チャンスを見つけてやってみてみたい。これはもうリアルタイムとしては苦痛を伴うため、参考程度にしておきたい。いずれ別方向、レンダリングに関する研究をするならば、それはそれでしっかりサポートをしていきたい。

## ポリゴナイザ解析

それでは、そろそろリアルタイムポリゴナイザについて研究をしていこう。順序よく説明していくが、一発でこれらのアルゴリズムに到達しているわけではない。また危険なのは、段階的に開発しているの、ある段階で致命的なアルゴリズムであっても、気がつかなければそのまま引きずっていることになっていることである。もうひとつ恐ろしいのはこれを読むことによって、皆がほかのアルゴリズムを考えなくなることである。思いつくだけのアルゴリズムを示し、それぞれの利点、欠点を洗い出していきたいと思う。あくまでも一例として参考にするだけが望ましい。

また、きわめて苦勞するのは、個別の処理系ごとに分けて扱ってはいけないうことである。ある処理が終わった段階で、扱っていた変数の最終値が次のステップで継続して使用できたりすることがあるので、本来はこんな単純な考えでは到達できない。いかに全体を把握するか、うまい閃きを炸裂させるか、がコーディングの秘訣である。だが最大の要素として運があることも忘れてはならない。貴方がコーディングでハマるのは、数あるアルゴリズムの中から運悪くハマるアルゴリズムを想定してしまったからなのである。

## 単純な水平ラインを考える

まず塗り潰された多角形を描画することを考える。グラフィックツールの大部分はシードフィル法をとっている。ペイントという動作は、扱いやすいし、多角形に鈍角が交じっていても処理できる。これはいくらかでもアルゴリズムがあるが、処理の複雑さ、効率の問題からリアルタイム制御にはまず向かない。まずこれはポリゴナイザの候補から外す。

そこで一般的なものを考えるとやはりソリッドスキャンコンバージョンである。まず大きな意味合いで考えよう。図1を見てほしい。水平方向のラインでポリゴンの最下点まで調べる、つまりスキャンするのである。そのスキャンラインとポリゴンとの交点がエッジである。ポリゴンすべての頂点が鋭角であれば交点は必ず2つ現れる。その間を描画するだけである。つまりこれは各左右エッジ間の水平ラインをY方向にループすればよい。

ではコーディングに向けてもう少し突っ込んでいこう。アセンブラ使いならこの水平ラインループは展開するのが当たり前。まずこれによってX方向のループは外すことができるのである。本来はさらにY方向のループも外しておきたいのだが、それについてはまたいずれ考えることにする。

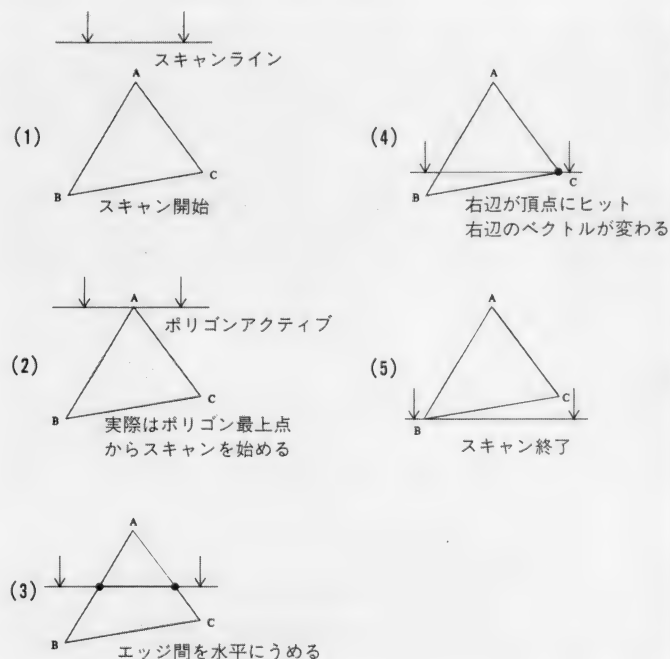
ではどのように展開すればよいだろうか。まず2ドットのアクセスはどうする？ 当然move.wを2つ並べるよりmove.lが高速。ではさらに16ドットのアクセスはどうだろう。これはmove.lを8個並べるよりもmovem.lを使用する。当然ここで扱うレジスタはすべてカラーコードで埋める。1ドット1ワードなので、各レジスタの上位、下位ワードともに同じカラーコードを保存する。ここで問題なのはmovem.lがレジスタを多数必要とすることであろう。X68000の場合、このグラフィックがネックとなっているため、できる限りレジスタを余らせてこの方法に臨みたい。

## エッジ検出を考える

次の基本となるエッジ検出を考える。これはポリゴンを描画するときにスキャンライン順に処理していくからである。簡単な2つのアルゴリズムを挙げて考える。

まず共通する内容を確認するが、これは決して線分を作るためのものでなく、1ラスタごとのエッジを検出するだけである。よって図2のようにエッジが画面上で連続する必要がないのである。

図1 スキャンラインの様子





## ハードコア3Dエクスタシー(第2回)

## I : Bresenhamのアルゴリズム

代表的な方法としてBresenhamのアルゴリズムがある。あまりに有名なため具体的な内容は示さないことにする。一応ここで使用できる例としてリスト1を示す。一般的にはライン描画ルーチンで使われている奴だが、連続する必要がない以上やや特殊なルーチンになっている。まずこの方法では1ループで1以上の差分が許されていないため、こゝである処理を行わねばならない。具体的には誤差項の符号が変化するまでループを組まなければならない。また、面倒なのはラインを描画する方向により処理

## ■リスト 1 Bresenhamによるエッジ検出

```

2:
3:
4:         .include          IOCSCALL.MAC
5:         .include          DOSCALL.MAC
6:
7: moveq.l #12,d1
8: IOCS     _CRTMOD          *512*512
9: IOCS     _G_CLR_ON        *クリアオン
10:
11: bsr      SUPER            *スーパーバイザー
12: bsr      BRZEM            *メインルーチンへ
13: bsr      USER            *ユーザーモード
14: DOS      _EXIT            *終わり
15:
16: SUPER:
17:         suba.l    a1,a1
18:         IOCS     _B_SUPER
19:         move.l    d0,sspsave
20:         rts
21:
22: USER:
23:         move.l    sspsave,a1
24:         IOCS     _B_SUPER
25:         rts
26:
27: sspsave:
28:         dc.l      0
29:
30: $
31: *=====
32: *エッジ専用ブレゼンハムラインルーチン
33: *=====
34: BRZEM:
35:         move.w    #256*2,d0      *始点X座標256(*2はアドレス値)
36:         lea.l     $c40000,a0     *始点Y座標256
37:         move.w    YPOS,d7        *Y方向長さ
38:         move.w    XPOS,d6        *X方向長さ
39:
40:         move.w    d7,d5          *傾きを表す値
41:         move.w    d6,d4          *
42:         lsr.w     #1,d3          *傾き項/2
43:         sub.w     d6,d4          *初期値のために引いておく
44:
45: loop:
46:         bsr      PUT              *エッジを求めたための描画へ
47:
48:         add.w     d6,d4          *aを加える
49:         bmi     next            *負ならばa'を引かない
50:         addq.w    #2,d0          *X方向移動
51:         sub.w     d5,d1          *係数a'
52:         bcc     brz1            *傾き項が正ならさらにループ
53:
54:         next:
55:         lea.l     $400(a0),a0    *1ライン下へ
56:         dbra     d7,loop         *Y方向ループ
57:         rts
58:
59: PUT:
60:         move.w    #$ffff,0(a0,d0,w) *エッジ表示
61:         rts
62:
63: YPOS:    dc.w     100            *Y1(座標ではない)
64: XPOS:    dc.w     211            *X1
65:
66: $
67: * テストとしてXPOS, YPOSを適当に変えて動かしてみよう。
68: * エッジが切れないが連続している様子はつかめると思う。
69:
70: $
71: * PUTルーチンでD0レジスタを使用しているのは、本来左右エッジの
72: * 座標を求め、その間を塗り潰すだけのルーチンにしなければならないか
73: * であること、いってもわからないと思うが、全体像から考えればこの
74: * 方法が断言良いのではこうしてあるだけ。
75:
76: $
77: * 途中で出てくるaはレジスタを使用しているが、これらはスタックに
78: * 積んでおくのが正しい。理由は本文を参考にしてもいい。
79:
80: $
81: *
82: *
83: *
84: *
85: *
86: *
87: *
88: *
89: *
90: *
91: *
92: *
93: *
94: *
95: *
96: *
97: *
98: *
99: *
100: *
101: *
102: *
103: *
104: *
105: *
106: *
107: *
108: *
109: *
110: *
111: *
112: *
113: *
114: *
115: *
116: *
117: *
118: *
119: *
120: *
121: *
122: *
123: *
124: *
125: *
126: *
127: *
128: *
129: *
130: *
131: *
132: *
133: *
134: *
135: *
136: *
137: *
138: *
139: *
140: *
141: *
142: *
143: *
144: *
145: *
146: *
147: *
148: *
149: *
150: *
151: *
152: *
153: *
154: *
155: *
156: *
157: *
158: *
159: *
160: *
161: *
162: *
163: *
164: *
165: *
166: *
167: *
168: *
169: *
170: *
171: *
172: *
173: *
174: *
175: *
176: *
177: *
178: *
179: *
180: *
181: *
182: *
183: *
184: *
185: *
186: *
187: *
188: *
189: *
190: *
191: *
192: *
193: *
194: *
195: *
196: *
197: *
198: *
199: *
200: *
201: *
202: *
203: *
204: *
205: *
206: *
207: *
208: *
209: *
210: *
211: *
212: *
213: *
214: *
215: *
216: *
217: *
218: *
219: *
220: *
221: *
222: *
223: *
224: *
225: *
226: *
227: *
228: *
229: *
230: *
231: *
232: *
233: *
234: *
235: *
236: *
237: *
238: *
239: *
240: *
241: *
242: *
243: *
244: *
245: *
246: *
247: *
248: *
249: *
250: *
251: *
252: *
253: *
254: *
255: *
256: *
257: *
258: *
259: *
260: *
261: *
262: *
263: *
264: *
265: *
266: *
267: *
268: *
269: *
270: *
271: *
272: *
273: *
274: *
275: *
276: *
277: *
278: *
279: *
280: *
281: *
282: *
283: *
284: *
285: *
286: *
287: *
288: *
289: *
290: *
291: *
292: *
293: *
294: *
295: *
296: *
297: *
298: *
299: *
300: *
301: *
302: *
303: *
304: *
305: *
306: *
307: *
308: *
309: *
310: *
311: *
312: *
313: *
314: *
315: *
316: *
317: *
318: *
319: *
320: *
321: *
322: *
323: *
324: *
325: *
326: *
327: *
328: *
329: *
330: *
331: *
332: *
333: *
334: *
335: *
336: *
337: *
338: *
339: *
340: *
341: *
342: *
343: *
344: *
345: *
346: *
347: *
348: *
349: *
350: *
351: *
352: *
353: *
354: *
355: *
356: *
357: *
358: *
359: *
360: *
361: *
362: *
363: *
364: *
365: *
366: *
367: *
368: *
369: *
370: *
371: *
372: *
373: *
374: *
375: *
376: *
377: *
378: *
379: *
380: *
381: *
382: *
383: *
384: *
385: *
386: *
387: *
388: *
389: *
390: *
391: *
392: *
393: *
394: *
395: *
396: *
397: *
398: *
399: *
400: *
401: *
402: *
403: *
404: *
405: *
406: *
407: *
408: *
409: *
410: *
411: *
412: *
413: *
414: *
415: *
416: *
417: *
418: *
419: *
420: *
421: *
422: *
423: *
424: *
425: *
426: *
427: *
428: *
429: *
430: *
431: *
432: *
433: *
434: *
435: *
436: *
437: *
438: *
439: *
440: *
441: *
442: *
443: *
444: *
445: *
446: *
447: *
448: *
449: *
450: *
451: *
452: *
453: *
454: *
455: *
456: *
457: *
458: *
459: *
460: *
461: *
462: *
463: *
464: *
465: *
466: *
467: *
468: *
469: *
470: *
471: *
472: *
473: *
474: *
475: *
476: *
477: *
478: *
479: *
480: *
481: *
482: *
483: *
484: *
485: *
486: *
487: *
488: *
489: *
490: *
491: *
492: *
493: *
494: *
495: *
496: *
497: *
498: *
499: *
500: *
501: *
502: *
503: *
504: *
505: *
506: *
507: *
508: *
509: *
510: *
511: *
512: *
513: *
514: *
515: *
516: *
517: *
518: *
519: *
520: *
521: *
522: *
523: *
524: *
525: *
526: *
527: *
528: *
529: *
530: *
531: *
532: *
533: *
534: *
535: *
536: *
537: *
538: *
539: *
540: *
541: *
542: *
543: *
544: *
545: *
546: *
547: *
548: *
549: *
550: *
551: *
552: *
553: *
554: *
555: *
556: *
557: *
558: *
559: *
560: *
561: *
562: *
563: *
564: *
565: *
566: *
567: *
568: *
569: *
570: *
571: *
572: *
573: *
574: *
575: *
576: *
577: *
578: *
579: *
580: *
581: *
582: *
583: *
584: *
585: *
586: *
587: *
588: *
589: *
590: *
591: *
592: *
593: *
594: *
595: *
596: *
597: *
598: *
599: *
600: *
601: *
602: *
603: *
604: *
605: *
606: *
607: *
608: *
60
```

を分けなければならないことである。これはエッジ検出である以上、Y方向は固定なので単に正負のみで処理を分ける。

そしてコーディングの段階であるが、まず必要となるパラメータが多すぎる。先ほどの水平ラインと密接に絡むため、パラメータにレジスタを食われすぎるとかなり危険である。ラインと違って左右のエッジを同時に算出するため、普通のルーチンの倍の手間がかかる。定数はスタックに積むとしても、レジスタに残すべき変数は座標、誤差の2つであろう。誤差もスタックに積んでもかまわないが、レジスタより遅い。また条件判定、条件ループを伴うため速度は不安定であり、条件はあまりよくない。

さらに先のレベルの話になるが、SLASHシステムを全体から見たときには、より致命的な欠点をもっているのである。これは別の機会に紹介しよう。

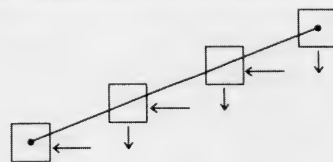
## II：16ビット固定小数点演算

では上記の欠点をカバーし、かつ簡潔にコーディングできる方法を考えよう。一般的に、コンピュータにとって小数は厳しい演算といわれている。まず10進数で考えるとかなり致命的である。16進数で考えれば、というよりもっと単純に考えれば小数とは極めて単純なものである。たとえば1ロングワードであるレジスタを考える。これを上位、下位それぞれ1ワードを整数部、小数部にあててやれば非常に楽に小数が扱える。MPU68000では、都合がいいことにSWAP命令が用意されている。これで小数を含んで計算し、SWAP一発で整数部が取り出せるのである。サンプルとしてリスト2を示しておく。

この16ビット固定小数点演算はこれだけに限らず、あらゆる制御にかなり有効である。たとえばゲームであればキャラクタの座標管理である。加速度を持つ運動をさせるのに時分割でテーブルから移動量を取ってくるなんてのよりも速度パラメータに加速度パラメータを毎回足して、速度を座標に加えるだけ。小数を導入すれば滑らかな運動が簡単に行える。まあ参考程度にしてほしい。

話を本題に戻そう。これでどのようにエッジを検出するかはもうわかるはずだ。図3を見てほしい。当然このXLは符号をもっている。だからループでは毎回座標にXL/YLを加算するだけで次のエッジを

図2 エッジの検出方法



エッジはラインではない。Y方向1ドット  
ずつのX座標さえ求められればよい



得ることができる。必要なパラメータは座標、そしてXL/YLだけである。後者は定数が1つ、変数は座標のみであり、レジスタもかなり余裕がある。

## 任意のポリゴンへの対応

以上でエッジ検出は理解できた。ではエッジを検出させるために必要なことを考えなければならない。いかにして多角形であることを認識するか？つまりポリゴンの頂点からどれをどのようにパラメータとして渡すか、である。ここはかなり難易度が高く、いくらかでも方法はある。また先ほどのエッジ部でもより有効な方法があり、どんなときでもこれこそベストである、という方法は存在しない。

また、ここのアルゴリズムしだいでデータ構造までも考えておくべきであろう。ポリゴンデータをヘタに想定するとここもかなりハマる領域である。説明するのだけがすべてではない、決して固まることのないよう、柔軟に対処すべし。

### I：ベクトル判定法（オリジナル？）

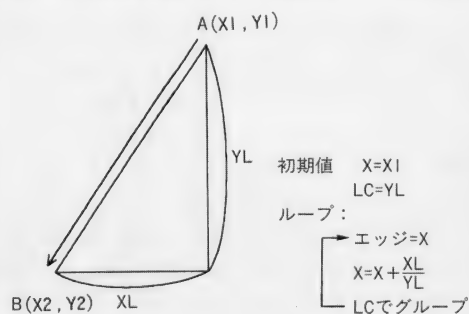
これは多角形をスキャンライン順に処理することを利用している。簡単な手順を以下に示す。

- 1：2D頂点をY順にソートする
- 2：頂点の状態によって処理を分ける
- 3：ベクトルにより左右エッジ決定
- 4：各辺のベクトルを計算する
- 5：ベクトルを元にスキャンライン順に最下点まで描画する

なにかいいいかまったく理解できないと思うので、以下図4を参考にしつつ具体的に考えることにする。図4では例として三角形を挙げている。

まず頂点のソートであるが、当然スキャンラインで処理するためにこうしているのである。ここで問題点が1つ。頂点が多くなれば当然ソートは重くな

図3 固定小数点を使ってエッジ検出



コーディングの際、Xの1ドットを10000<sub>H</sub>で表す

たとえば  $XL:YL$ が 1:2ならば  $\frac{XL}{YL}$ は8000<sub>H</sub>

$XL:YL$ が -2:1ならば  $\frac{XL}{YL}$ は-20000<sub>H</sub>

る。SLASHシステムでは、実用性を考え四角形まで処理できるようにしてある。五角形以上のポリゴンは、あまり一般的だとは思えないので問題ないとしている。4点のみのソートなら単純比較で十分可能であり、当然SLASHシステムではそうしている。

次に頂点の状態によって処理を分けるとある。エッジのベクトルが変化するのにはスキャンラインが頂点のどれかに重なったときである。このときの頂点の状態を調べるのである。リスト2に示したように、小数値を頂点に重なるたびに覚えてやればよいのである。具体的には、左右エッジのベクトル（小数値）、次の頂点が現れるまでのループカウンタをエッジ検出+表示ルーチンに与える。そのあと、たとえば左側のエッジが頂点に重なったのであれば左エッ

### ■リスト2 固定小数点を使ったエッジ検出

```

1:
2:
3:      .include      IOCSCALL.MAC
4:      .include      DOSCALL.MAC
5:
6:      moveq.l #12,d1
7:      IOCS      _CRTMOD      *512*512
8:      IOCS      _G_CLR_ON    *クリアオン
9:
10:     bsr      SUPER          *スーパーバイザー
11:     bsr      MI16           *メインルーチンへ
12:     bsr      USER          *ユーザーモード
13:     DOS      _EXIT          *終わり
14:
15: SUPER:
16:     suba.l    a1,a1
17:     IOCS      _B_SUPER
18:     move.l    d0,sspsave
19:     rts
20: USER:
21:     move.l    sspsave,a1
22:     IOCS      _B_SUPER
23:     rts
24: sspsave:
25:     dc.l      0
26:
27: *
28: *=====
29: *16ビット固定小数点ラインルーチン
30: *=====
31: MI16:
32:     move.l    #256*10000,d0 *始点X座標256(下位は小数)
33:     lea.l     $c40000,a0    *始点Y座標256
34:     move.w    YPOS,d7
35:     move.w    XPOS,d6
36:
37:     move.l    #57fff,d5      *0.5ドットに相当
38:     divu.w    d7,d5          *0.5/YL
39:     muls.w    d6,d5          *0.5*XL/YL
40:     add.l     d5,d5          *XL/YL
41:
42: loop:
43:     bsr      PUT              *エッジを求めたので描画へ
44:     add.l    d5,d0
45:     lea.l    $100(a0),a0     *1ライン下へ
46:     dbra     d7,loop         *Y方向ループ
47:     rts
48:
49:
50: PUT:
51:     move.l    d0,d1          *エッジ表示
52:     swap.w    d1
53:     add.w     d1,d1
54:     move.w    $ffff,0(a0,d1.w)
55:     rts
56:
57:
58: YPOS:     dc.w    100        *YL(座標ではない)
59: XPOS:     dc.w    211        *XL
60:
61: *
62: *
63: * 37行から4行がXL/YLを算出する部分。もしYLがそれほど
64: * 大きくなければ先に57FFFをYLで割ることで計算を軽減
65: * することができる。このことにより誤差が大きくなる場合は別の処理
66: * にしなければならないが、ここでは扱わないことにする。
67: * プレゼンハムのルーチンと違い、初期設定は比較的前端からわかるが
68: * 全体として効率を望むならばこちらが有利であると手前している。ま
69: * たこのリストではさらなる順位を示すことが出来ないが追って説明し
70: * てゆくことにする。
71:
72:

```



# ハードコア3Dエクスタシー(第2回)

図4-1 エッジ検出の基本動作

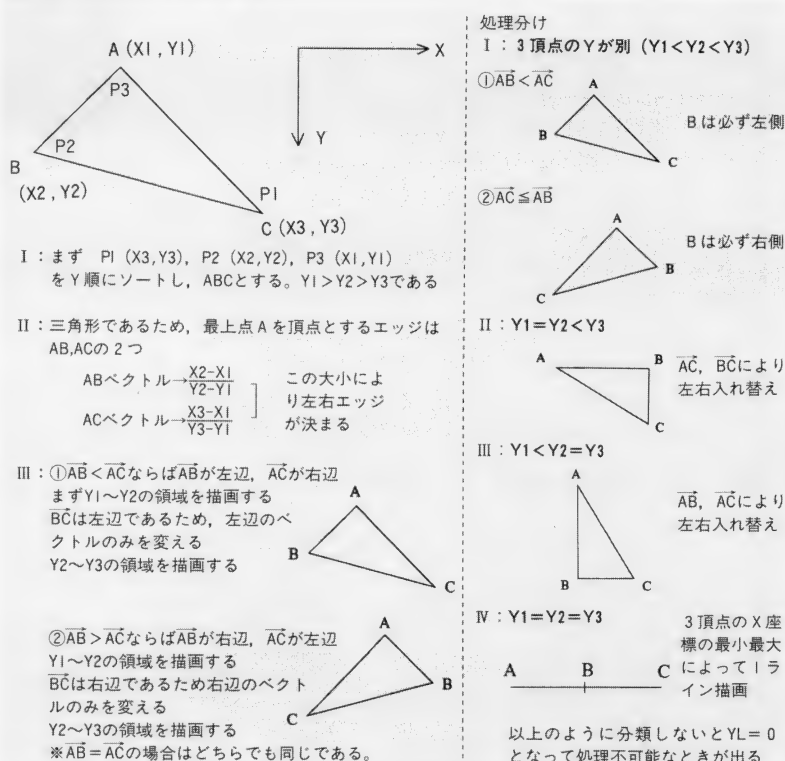


図4-2 描画段階での動作

描画ルーチン:

パラメータ: 左辺ベクトル  
右辺ベクトル  
YL (ループカウンタ)  
始点X座標 (左)  
始点X座標 (右)  
始点Y座標

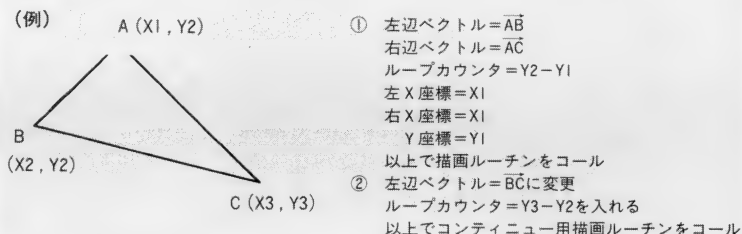
これにより描画を開始する。Y座標によって、同一ライン左はしのアドレスを計算し、ループカウンタによって1ラインずつスキャンしていく

リターンするとき残すパラメータは以下の通り

- ・左辺ベクトル
- ・右辺ベクトル
- ・X座標左
- ・X座標右
- ・最終スキャンラインのY座標が示す描画アドレス

さらに頂点にヒットしたときのために必要なパラメータを残してループに入る

実際にはリターンしてきた各パラメータはレジスタに入っているため、左右のベクトルのみを変更してさらにコンティニュー用描画ルーチン呼び出す



ジのベクトルを新たに指示してさらにループ、以下これをポリゴン最下点まで続けるわけである。とすれば、各辺のベクトルとは何を示すかはわかると思う。これはいわゆるXL/YLである。本来ベクトルとはいいがたいのだが、あえてベクトルと呼ばせしめよう。Y1ラインに対するXの移動量であることをしっかり理解してもらいたい。これをループ先頭で指示してやればエッジを計算してくれることになる。

頂点までループして次のエッジのベクトルを与えるわけだが、判定された頂点が左右どちらのエッジのものかをどうやって判定するのであろうか。これはいままでの内容を踏まえたうえで考えてみる。

まず最上点から左右の辺を決定する方法を考えてみる。これは単純に先ほどのベクトルの大小で決定できるのである。画面座標系で見ればベクトルの小さいほうが左辺、大きいほうが右辺であることは明白である。次の頂点からであるが、これは破綻が生じないために必要なベクトルを求め、それによってあらかじめ左右どちらかを決定しておいてから、まとめてループを呼べばかなりうまくいく(図4)。

さて、これはこれでかなりごまかしのある方法である。まず絶対条件として各頂点が180度以下であることが絶対である。そうでないポリゴンは左右エッジ1つずつでは処理できないため、このアルゴリズムではまともに表示できない。まともに、というのがこの方法の利点である。逆に見れば、四角形ならこのアルゴリズムはどのような4点でもエラーを起こさずにうまく処理してくれるのである。正しいデータで起動すれば平面上の四角形を形成しないことはなさそうだが、絶対になんらかの誤差により結果は美しい状態になる。与えられる座標が誤差を含む以上、どんな4点でもそれなりに動くことはシステムの信頼性としては重要である。

## 予告

さて、区切りの悪いところで切ってしまったがこのアルゴリズムはまだ先がある。よって来月はこの応用、さらに不都合の生じる状態を考察する。そしてアクティブエッジソートやそのほかのアルゴリズムを続けて研究していきたい。

それにしても、今月は酷いスケジュールであった。締め切り前後10日は仕事は一切できない状態となり、かなりあちこちの人に迷惑をかけてしまった。最後に、タキシードの情報を教えてくださった方、ありがとう。貴方のいうそれこそ俺の求めていたアレであった。あとドラキュラの図を描いたのは俺ではない。俺はリアリティを追求する。ということで車がくるからもうウハウハ。死んでなければまた来月。



# バックナンバー案内

ここには1992年11月号から1993年10月号までをご紹介しました。現在1992年6、7、9、12、1993年6～10月号の在庫がございます。バックナンバーはお近くの書店にご注文ください。定期購読の申し込み方法は156ページを参照してください。

1992

1993



## 11月号 (品切れ)

特集 ゲームマネージメント

DoGA CGアニメーション講座/大人のためのX68000  
響子 in CGわーど/ショートプロ/よいこのSX-WINDOW  
ハード工作/ANOTHER CG WORLD/Computer Music入門  
●新製品紹介 CHART PRO-68K  
LIVE in '92 ストリートファイターII/スーパーマリオ 他  
THE SOFTOUCH キャッスルズ/シュートレンジ/  
ポピュラスII/サンダーレスキュー  
全機種共通システム 実践Small-C講座(7)EDIT



## 12月号

Oh!X 5周年特別企画 ショートプロ大集合

DoGA CGアニメーション講座/マシン語プログラミング  
響子 in CGわーど/ショートプロ/よいこのSX-WINDOW  
大人のためのX68000/ハード工作/Computer Music入門  
●エレクトロニクスショー'92  
LIVE in '92 LAST CHRISTMAS/闇の血族/ユフォーリー  
THE SOFTOUCH デスブレイド/ムーンクレスタ&テラクレスタ/  
ふしぎの海のナディア/ロードス島戦記II 他  
全機種共通システム 実践Small-C講座(8)MAKE



## 1月号 (品切れ)

特集 D.I.Y.ハードウェア

DoGA CGアニメーション講座/マシン語プログラミング  
響子 in CGわーど/ショートプロ/よいこのSX-WINDOW  
大人のためのX68000/ハード工作/Computer Music入門  
●新製品紹介 サンダーワード/SX広辞苑  
LIVE in '93 ムーンライト伝説/チャコの海岸物語  
THE SOFTOUCH オーバーテック/スライダー飛竜/  
エアーマネジメント/バイブドリーム 他  
全機種共通システム 実践Small-C講座(9)EDC-Tの拡張



## 2月号 (品切れ)

特集 画像創造のために

DoGA CGアニメーション講座/マシン語プログラミング  
響子 in CGわーど/ショートプロ/よいこのSX-WINDOW  
ハード工作/吾輩はX68000である/Computer Music入門  
●新製品紹介 Communication SX-68K  
LIVE in '93 FIRE CRACKER/サンバDEグワッシャ!  
THE SOFTOUCH 種/ドラゴンスレイヤー英雄伝説/  
機甲装神ヴァルカイザー/キングス・ダンジョン  
全機種共通システム BLACK JACK



## 3月号 (品切れ)

特集 X-BASICを学ぶ

DoGA CGアニメーション講座/マシン語プログラミング  
響子 in CGわーど/ANOTHER CG WORLD/ハード工作  
ショートプロ/Computer Music入門/280's Bar  
●緊急速報 32ビットマシンX68030  
●新製品紹介 音源モジュールSC-33/GS音源搭載JW-50  
LIVE in '93 ストリートファイターII/晴れたらいいね 他  
THE SOFTOUCH 究極タイガー/チェルノブ/シムアント 他  
全機種共通システム シューティングゲームコアシステム作成法(1)



## 4月号 (品切れ)

特集 X68第7世代へ

DoGA CGアニメーション講座/マシン語プログラミング  
響子 in CGわーど/ショートプロ/よいこのSX-WINDOW  
ハード工作/吾輩はX68000である/Computer Music入門  
●決定! 1992年GAME OF THE YEAR  
●名作ゲーム再遊記  
LIVE in '93 FIGHTMAN/ミンキーモモより 愛しのマシーンカ  
THE SOFTOUCH スターフォース/元朝秘史 他  
全機種共通システム シューティングゲームコアシステム作成法(2)



## 5月号 (品切れ)

特集 襲撃! SX-WINDOW

第8回 言わせてくれなくちゃだわ

DoGA CGアニメーション講座/ANOTHER CG WORLD  
響子 in CGわーど/ショートプロ/大人のためのX68000  
ハード工作/吾輩はX68000である/Computer Music入門  
●X68030へのソフトウェア対応について  
LIVE in '93 MAGICAL SOUND SHOWER/もう笑うしかない 他  
THE SOFTOUCH エトワールプリンセス/メガロニア 他  
全機種共通システム シューティングゲームコアシステム作成法(3)



## 6月号

創刊11周年特別企画 確率遊技シミュレーション

DoGA CGアニメーション講座/こちらシステムX探偵事務所  
響子 in CGわーど/ショートプロ/大人のためのX68000  
ハード工作/吾輩はX68000である/Computer Music入門  
●新製品紹介 SC-55mk II  
LIVE in '93 ストリートファイターIIより 春麗のテーマ/  
BAY YARD/LOVE&CHAIN  
THE SOFTOUCH 銀狼伝説/信長の野望-覇王伝 他  
全機種共通システム REVERSI



## 7月号

特集 席卷するローテク文明

DoGA CGアニメーション講座/こちらシステムX探偵事務所  
響子 in CGわーど/ショートプロ/マシン語プログラミング  
ハード工作/吾輩はX68000である/Computer Music入門  
●新製品紹介 ドローイングバット33070&MATIER  
LIVE in '93 Midnight Circle/今日の日はさようなら/赤い靴  
THE SOFTOUCH 悪魔城ドラキュラ/リブルラブル/大航海時代II/  
銀河英雄伝説III/幻影都市/ヴェルズナーク戦乱  
全機種共通システム MSX用S-OS "SWORD"



## 8月号

特集 C言語実践入門

DoGA CGアニメーション講座/こちらシステムX探偵事務所  
響子 in CGわーど/Computer Music入門/大人のためのX68000  
吾輩はX68000である/ショートプロ/ANOTHER CG WORLD  
●特別企画 夏真っ盛り、アマチュアリズムのX68000  
LIVE in '93 SPLASH WAVE  
THE SOFTOUCH 悪魔城ドラキュラ/リブルラブル/銀狼伝説/  
ロボットコンストラクションR.C./Winning Post  
全機種共通システム MACINTOSH-C再掲載



## 9月号

特集 光学式磁気円盤MO

DoGA CGアニメーション講座/こちらシステムX探偵事務所  
響子 in CGわーど/ショートプロ/大人のためのX68000  
ハード工作/Computer Music入門/ANOTHER CG WORLD  
●新製品紹介 OS-9/X68030  
LIVE in '93 ファイナルファンタジーVのテーマ/銀河鉄道999/  
アルスラーン戦記IIより 汗血公路/ちやうちょ  
THE SOFTOUCH 悪魔城ドラキュラ/コットン/ダーク・オデッセイ 他  
全機種共通システム 7並ベ/SLANG再々掲載



## 10月号

特別企画 秋祭りPRO-68K

ハードコア3D/Computer Music入門/マシン語プログラミング  
DoGA CGアニメーション講座/こちらシステムX探偵事務所  
響子 in CGわーど/ショートプロ/吾輩はX68000である  
●特別付録 秋祭りPRO-68K (5"2HD)  
●SCSIバックンTOWER JACK  
LIVE in '93 未来予想図II/OutRunより PASSING BREEZE  
THE SOFTOUCH コットン/The World of X68000/あにまーじゃんV3  
全機種共通システム シューティングゲームコアシステム作成法(4)



# 新しい世界へ静かに発進

Ogikubo Kei 荻窪 圭

昔話をはじめほど老化現象の進んだ私であるが、つらつらと思ひ返すに、なんでこの連載がはじまったか、っていうと、編集長にビジネスソフトを使って何かする連載でもやってみないかといわれたのがはじまりだったような気がする。それを平たくいえば「Kamikaze復権」だったわけだ。

KamikazeはX68000が出たばかりの頃、最初の本格的ビジネスソフトとして登場したスプレッドシートであった。いまはもうWindowsとかMacintosh上に、Excel 4.0やら1-2-3/WやらQuattro Proやらがあってどんどん取り残されてしまったわけであるが、5年前となると、Windowsなどひとさまに見せできる状態になく、せいぜい高価なMacintosh用にExcel 2.2があったくらいで、それはもう、Kamikazeは筋のいい、問題点はいっぱい抱えていたけれども、志の高いソフトであった。結局、Kamikazeを超える筋のいい、志の高いソフトって出なかった気がする。

で、「連載かあ」とつぶやきながら当時編集部のあった九段下への道を歩いていると「大人のためのX68000」という言葉が降りてきて、この連載がはじまったのである。

内容はタイトルがひっぱっていつてくれるはずであった。

## X 大人のための

このタイトルにはさまざまな意味が込められている。そのうちのひとつには、過ちを繰り返さないように、という意味がある。

かつて、MZシリーズは多くのユーザーを育てた。多くの筋のいいパワーユーザーを育てた。MZシリーズだからこそよい方向に育った、って面は多かったはずだ。

MZシリーズで育ったユーザーはどこへ行ったか。誰も歳をとれば大人になる。大人になったMZユーザーはMZを捨て、あるいは卒業し、多くが止むを得ずかどうか好きこのんでか98へ行ってしまったのである。98ソフトを支えたプログラマには、MZ出身者が少なくなかったはずだ。つまり、MZは育てるだけ育て、育ったところをNECにかっさらわれたのである。大人になり、ワープロも使い、ビジネスもし、と世間のしがらみのなかにあって、MZはあまりに非力だったのだ。

X1シリーズも筋のいいユーザーを育てた。成長したユーザーは16ビットパソコンを望み、ちょうどよいタイミングでX68000が登場した。X68000も筋のいいユーザー

1990年9月号にはじまったこの連載も、とうとう最終回を迎えることとなりました。3年ちょっとのあいだに大きく変わったパソコンそしてX68000をとりまく状況をふり返りつつ、いろいろ考えてみましょうか。

を育てた。そのユーザーが歳をとっていったとき、X68000から離れていかにするにはどうしたらいいか。いままで弱かった実用系のソフトもちゃんとサポートしていかなければならんのやないか。でも、98用ビジネス活用誌みたいなことはしたくない。じゃあ、実用系のソフトでめいっぱい遊んでやろう。とまあ、そういうわけである。

当初は、Kamikazeに続き、ワープロソフトやらデータベースやらドローイングやらいろいろ出るはずで、一度出たソフトもどんどんバージョンアップしていくはずであった。

しかし、そういうことはなく、荻窪圭はその間どんどん墮落し、腐っていった。相変わらず98もDOSもWindowsも好きではないけど。他人が腐るのを見るのは耐えられないが、自分が腐るのはけっこう気持ちいい。世の中えてしてそういうものである。ずぶずぶずぶ。

歳のとり方には2つあると最近思う（思うだけなら簡単）。「枯れていく」か「腐っていく」かだ。どうせなら、腐るんじゃなくて枯れたいものである。

## X ネコとカマキリ

明け方に、原稿を書いているとうちのネコが窓の外を見て啼くわけで、外へ出たいのだなあと思を明け放ってやると、漂ってくる外気の香りを必死になって嗅いでいる。私は窓を開けたままパソコンの前に戻る、ネコはいつの間にか外へ出ている。しばらくすると、うろうろしていたネコが部屋に飛び込んで来る。なんか銜えているようだ。カマキリであった。床に銜えてきたカマキリを落とし、爪で弄びながら何度も銜えたり放したりしている。面白いのでずっと見てみると、口が忙しく動いている。昆虫がつぶれるときの特有の音をさせてうまそうに食っていたのだ。ネコが満足げに水を飲みに行っただけには、上半身だけ残されたカマキリが弱々しく蠢いていた。

考えてみたら、ネコが虫を食うところってのはじめて見た。某編集者の家のネコはセミを好きこのんで食べていたらしい。

別に、ドナルド・フェイゲンの新譜がカマキリアドだとか、そういう話をするつもりではない。

Windowsを使っていると、キッチン質の殻をバリバリと噛み砕いているネコになった気がする、という話をするつもりはある。バリバリバリ。



## Xビデオとパソコン

Macintosh Centris660AVのVRAMの話をしてしよう。

Centris660AVは標準でビデオキャプチャ機能をもっている。ビデオキャプチャボードというのはたいてい、アナログビデオ信号をデジタル化し、その結果をアプリケーションに渡し、アプリケーションがそれを表示する。IBM PCになるとVGAがフューチャーバスってものをもっていて、それを使ってウンヌンカンヌンって話にもできるようなのだが、そのへんはよくは知らない。

で、上記の方法だと、映像出力はパソコンのグラフィックの描画速度に左右されるし、CPUへの負担も大きい。だから、テレビを見ながらパソコンする、という用途には向いていない。もしテレビを見ながらパソコンしたいなら、ビデオ表示用の画面と通常のパソコンのグラフィック画面の2画面をもち、パソコンの画面に窓を開けて、その向こうでビデオ表示専用画面にビデオ映像を映すのがてっとりばやいわけだ。合成をハードでやってしまうわけ。

Centris660AVはそれを行っている。実現方法がまたユニークだ。

まず、このマシンはVRAMを1Mバイトもっている。X68000の倍だ。512×512ドットで16ビットカラーの画面を2つもつことができる計算だ。Macintoshでは縦横比4:3が原則であるから、640×480ドットで16ビットカラーが標準となる。そうすると、600Kバイトを消費する。400Kバイトは余るわけだ。いままでのMacintoshでは「余ったVRAMは誰にも知られずひっそりとお休みしていた」わけである。

そこで、Centris660AVでは640×480ドットでフルカラーのモードをもった。ここまではありがちだ。

面白いのは、ビデオキャプチャ時。VRAMをビデオ画像入力用に使うのである。で、ビデオ入力画像は640×480で16ビットカラー固定で、それは本体のVRAMを利用する。つまり、VRAMを2つに分けて、ビデオ入力用画面とパソコンのグラフィック画面にするわけだ。当然、VRAMは400Kバイトしか余らないから、640×480ドットで256色になる。16ビットカラーは使えない。だが、合成しているビデオ入力画面は16ビットカラーで表示されるのだ。けっこう面白い。このへんをハードでやっているから、ビデオ入力画像の表示は非常に高速だ。フルスクリーンでもウィンドウを切っても縦横比を変えても高速で気持ちがいい。

で、それをキャプチャすると、本体のグラフィック画面のほうに転送される。

いままでのMacintoshには見られない小技だ。でも、面白いよね。シャープが作りそうでしょ。

で、ビデオ入力画像の画質はどうかというと、S端子まで装備しているわりにはそんなによくない、っていうか、フルスクリーンでモニタをテレビ代わりにすると、どうしてもアラが出てしまう。なぜなら、NTSC信号は1フレームを偶数フィールドと奇数フィールドの2回に分けて送信しているのだが(つまり、秒間60フィールド)、そ

のうち、半分(つまり、奇数か偶数かのどちらかのフィールド)しかキャプチャしないからだ。それをフルスクリーン表示すると当然ながら、間引きしたような絵になる。

じゃあ、実際に秒間60フィールドをサポートしたビデオキャプチャボードはあるか、っていうと、ある。どれも高いけど、最近登場しはじめた。

次期X68000の話だが、そのくらいの機能は欲しいよね、っていうか、そういうAVな香りはやはり強く残してほしいよね。

そうそう、次期X68000だけど、いっそのこと、CPUをPowerPCにしちゃうってのは面白いんじゃないかと思ったりする。どうかなあ。で、いまX68000のソフトはエミュレーションで動かす。

家電みたいなデザインにして、最初から低音がガンガン鳴るステレオのスピーカーとアンプをもっている。スピーカーつたって、ちゃんとそこだけは密閉構造にして、少なくともCDラジカセ並みの音質はないとダメだな。CD-ROMドライブはもちろん内蔵する。ディスプレイテレビはやめて、その代わりに、チューナーは本体側でもつ。(一応)デジタルテレビだ。で、音楽CDをSCSIを通して直接サンプリングできたりするとなお楽しいだろう。

PowerPCはアップルーIBMーモトローラ連合の心臓となるプロセッサで、IBMとモトローラで共同開発し(もとの設計はIBM)、モトローラが製造するチップである。速いらしい。びゅんびゅんびゅん。

## Xとりあえず、さよなら

そういえば、この連載もいろいろと遊んだなあ。アンケート集計大会やったり、デジタルカメラで画像を入力してみたり(Hな画像撮り放題!), 写真データをもとにステレオ化に挑戦したり……。ほかになにやったっけ。覚えてないや(苦笑)。

そんなわけで、この連載も静かに終わる、ってことで、本当はEG Wordが出るまで続けたかったんだけど、それは縁がなかったこととしておこう。今月は非常に「大人ネタ」である「Photo CD」なんてのもあるが、これはレビューのページのほうで紹介する。それにしても、世の中でいちばん苦手なものが「ひとつのことを持続してやり続ける」という私が連載をした、ってことだけでも凄いよな。

X68000は、ほかのパソコンの急激なAV化(98でさえ、サンプリング音源やら256色やらフルカラーやらを搭載する時代になったのだ。恐ろしい)や急激な低価格化路線によって(おかげで、X68000+実用的セカンドマシンっていう2台構成もまた可能になっていったのだが)、立場がより明確になってしまった。簡単にいえば、アクション系ゲーム世界とオープンなアーキテクチャを利用した68000系の自由度の高いプログラミング世界とだ。私は残念ながらそのどちらの世界にもいない。連載を終えるいちばんの理由はそこにある。

そういうわけで、またどっかで会おうでしょう。

ご愛読ありがとうございました。



OS-9/X680x0

OS-9/X68030

# Ultra C & Professional Pack V1.1

## Technical Tool Kit V2.4.5

Nakamori Akira 中森 章

OS-9/X68030シリーズの第2弾、第3弾が発売されました。OS-9/X68030を紹介したのが9月号ですから、なかなか快調なペースでの製品発売でメーカーの意気込みが感じられます。活用方法が限られていたOS-9も、これでやっと本格的なプログラム開発に取り組めるようになります。

### Ultra C & Professional Pack V1.1

これは、Cコンパイラ、アセンブラ、リンカ、ソースレベルデバッグからなるコンパイラパッケージです。

Ultra Cコンパイラは、ANSIおよびISOの規格に完全合致と、すぐれた最適化機構を売り文句にしています。68000/68020/30/40/CPU32用のコードを出力することができ、OS-9/X68000でも使用可能です。

Ultra CはIコードと呼ばれる、CPUに依存しない中間コードを用いてコンパイル処理を行います。コンパイル開始時にC言語のソースプログラムを中間コードに変換し、複数のソースファイルを中間コードのレベルで1ファイルに結合して最適化します。この利点は、分割コンパイルのために複数に分けて書かれたソースプログラムをコンパイル時に1つのファイルとして処理できることです。ほかのファイルとの関数や変数の依存性を気にしなくてよいので、関数のインライン化や不要な変数の削除や定数の畳み込みなど高度な最適化が可能になります。中間コード方式のコンパイラの一般的な欠点は、多くの中間処理のために

コンパイル速度が遅いという点です。Ultra Cの性能をみるため、ドライストン2.1のプログラムをコンパイルしてみると、約1分かかります。Human68k上でGCCを使ってコンパイルする場合は、シャープ純正の遅いアセンブラを使用しても20秒程度です。しかし、これをどうみるかは個人の判断になるでしょう。最適化はプログラムのデバッグが終わった最終局面で行うという人にとっては、影響はないかもしれません。

ベンチマーク結果はUltra Cで約6,000ドライストン。Human68kのGCCよりもやや低いですが、まじめにマルチタスクしているOS-9上にしてみると結構よい値です。

動作モードは3つあります。compatモード、c89モード、uccモードです。compatモードは従来のマイクロウェアCとの下位互換です。出力するオブジェクトコードやコンパイルオプションにできる限り互換性をもたせてあります。このため、従来のMakefileなどをそのまま利用してコンパイルすることも可能です。最適化機構はUltra Cで向上していますから、従来資産を再コンパイルで性能向上できます。c89モードはANSI互換です。マイクロウェアCと同じく、このUltra Cもコンパイルオプションの名前が独特です。UNIXなどのCコンパイラに慣れていると多少違和感がありますが、このc89モードではUNIXとよく似たコンパイルオプションを使用できるようになります。uccモードはデフォルトの動作モードで拡張ANSIモードです。ANSI規格から

拡張した機能を使用できますが、現在はc89モードと大差ないようです。

### Technical Tool Kit V2.4.5

これは、OS-9/X68030でプログラミングをするための資料集で、OS-9/X68030専用です。内容はOSの内部やシステムコールを解説したドキュメントとデバイスドライバやシステムモジュールのソースファイルから構成されています。

付属の「RomBug」は、システムの拡張やI/Oドライバの拡張を支援するためのシンボリックデバッグです(写真1)。OS-9のシステムと独立しており、システム起動時に必要なコンソールやディスクのドライバもデバッグ可能ということです。システムの内蔵モジュールと差し替えて使用するため、組み込みにはシステムのジェネレーションが必要です。使用感覚はHuman68kのDBXのようでしたが、私自身はOS-9の内部を理解していないので実行時に何が起きているのかよくわかりませんでした。

このツールキットはかなり専門的です。システムを拡張しようと思っている人以外には必要がないかもしれません。

\* \* \*

今回も、マニュアルはオンラインマニュアルの形式で供給されています(写真2)。manコマンドで常時参照できるのは便利なのですが、マニュアルがなければディスク枚数(Ultra Cは7枚、Tool Kitは3枚)が半分以上になるので、インストール時間やディスク容量を考えると紙のマニュアルも捨て難いと思います。個人的には、オンラインと紙の両方が用意されていて、インストール時にオンラインマニュアルを読み込むかどうかを選択できるとうれしいのですが。ディスクの空き容量が10Mバイト必要というのも、ちょっとつらいですね。

■Ultra C & Professional Pack V1.1  
X680x0用 3.5+5"2HD版 45,000円(税別)  
■Technical Tool Kit V2.4.5  
X68030用 3.5+5"2HD版 20,000円(税別)  
マイクロウェアシステムズ ☎03(3257)9003

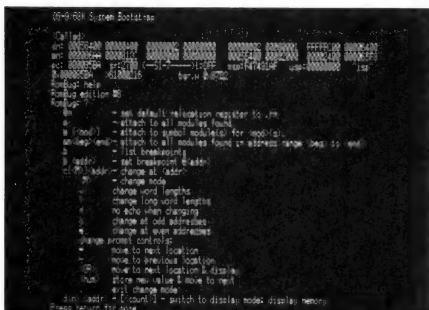


写真1 テンキーの「0」を押しながら起動すると現れるRomBugの画面。君は使いこなせるか？

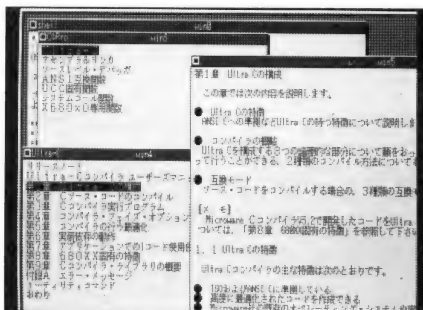


写真2 Ultra Cのオンラインマニュアル。ファイル容量は6 Mバイト。知りたいことが何でもわかる？



# Easydraw SX-68K

Tan Akihiko 丹 明彦

1993年8月号でβ版を紹介したSX-WINDOW用ドローツール「Easydraw SX-68K」の製品版レビューをお届けする。β版に見られたバグや変な仕様のほとんどが解消され、使えるツールに仕上がっている。SX-WINDOWもようやくDTP環境への第一歩を踏み出したのだ。

いまだからいってしまうが、β版を使ってみて、私はけっこう不安だった。ちょっとこのままじゃあ買えねえなあ、と思うつ、必携のツールとまでいいきっていた。で、発売されたいま、自信をもって、お買い得なソフトだと断言する。といいつつも、製品版が編集室に届くのを待ってきっちり動作確認をしてからショップに買いにいって小狡い私なのであった。

## Easydraw SX-68Kとは

SX-WINDOW向けのドローツールである。といっても、多くのX68000/030ユーザーには馴染みがないことだろう。

グラフィックツールは、ペイント系ツールとドロー系ツールに大別される。これまではグラフィックツールといえばペイントツールが大半であったが、このEasydraw SX-68Kの登場により、ドローツールもメジャーになることだろう。

ペイントツールは1ドットごとに色を決めることで絵を作り上げる。対してドローツールは、仮想的な紙の上に直線や三角形、円といった図形(ドローオブジェクトと呼ぶ)を置いていくことで図を作り上げる。

両者にはそれぞれ得手不得手があるが、本稿はドローツールのレビューであるから、ドローツールの長所を強調するような例を挙げよう。

Z'sSTAFF(いわずと知れたペイントツールである)で円を描いてみる。描いたあとで、大きさが気に入らなくなっても、大きさは変えられない。アンドウして改めて描き直すしかない。でもアンドウは無限には効かないのだ。続いて円の上に四角形を重ねて描いてみる。描いたあとで場所が気に

入らなくなっても、動かせない。移動コマンドを使えば、いまだで四角形があった部分の白く抜けてしまうだろう。

ドローツールは、こうした操作が得意である。つまり、描いた図形を拡大縮小したり、移動したりできる。図形は紙の上に置いてある部品であり、構造を持っている。画面に表示されているのは仮の見え方にすぎない。部品だから別の図に持って行って大きさを覚えて張りつけたりということも自由にできる。

ドローツールのうまみは、修正や再利用が楽であるということのほかに、印刷に極めて適しているということが挙げられる。一般的な傾向として、プリンタの解像度はディスプレイより高い。標準的なX68000の15インチモニタの解像度は約64dpi(1インチ=64ドット)、対してプリンタは普及率が高いローエンド製品でも180~360dpi。同じ面積でも画面よりプリントアウトのほうが高い表現力を持っていることになる。したがって、ペイントツールで描いた絵を印刷すると、ドットの粗さが目立つことになってしまう。さもなくば、仮想画面などを用いて、画面より遙かに大きなサイズの絵を描く必要があるのだ。

まとめると、ドローツールの長所は、

- ・描いた図形の修正が楽
- ・描いた図形の再利用が楽
- ・印刷が美しい

の3点に集約される。

## Easydrawの特徴

1) ドローオブジェクトとして、線分、長方形、円/楕円、多角形、ベジェ曲線、角の丸い長方形、扇形、スプライン曲線、それ

にテキストが扱える。

2) ドローオブジェクトのパラメータは多種多様で、もちろん一度描いたあとでも変えられる。線分は線の太さやラインスタイル(実線か破線か、破線の場合はそのパターン)、矢印の指定が可能。多角形などは輪郭線の太さや模様、または面の模様を指定できる。テキストはフォントの種類、サイズ、スタイル(イタリックや影つき白抜きなど)を文字単位で指定できる。

3) ドローオブジェクトのリサイズ(拡大縮小)や変形、回転などの編集機能がある(写真1)。

4) 複数のドローオブジェクトをグループ化して1個の疑似的なドローオブジェクトとして扱える。拡大縮小、回転は自由。もちろんいつでも個々のドローオブジェクトに分解できる。

5) マルチウィンドウ環境を生かして、ある図の一部をほかの図やキャンバス、X、シャープペン、Xなどにカット&ペーストすることができる。

6) たいていのプリンタに美しく印刷できる。ラインプリンタとして、(SX-WINDOWがサポートしている)シャープのCZ系列やエプソンのESC/P、キャノンのBJ-10シリーズなどをサポート。またEasydraw SX-68KにはSX-WINDOW用のレ

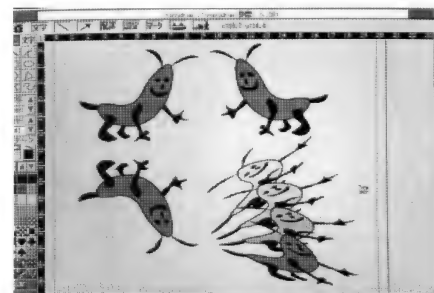


写真1 編集機能で回転や変形ができる



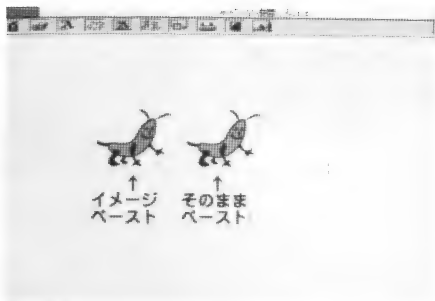


写真2 シャーペン.Xにペーストする

ユーザープリンタドライバが付属している。エプソンのESC/PageやキヤノンのLIPS-III、アドビシステムズのPostScriptに対応。

## 使い勝手はどうか

この手のツールにとって、ユーザーの操作に対してどういう挙動を示すかということはデザインの要である。どういうタイミングでどこをクリックしたりドラッグしたりしたら何が起こるかということが、使い勝手を決めてしまう。表現力を上げようとすればいろいろな動作モードが必要になるが、よく考えないで作ると操作が煩雑なだけのものになってしまう。Macintosh用ドローツール「Macdraw」の作法は、モード指向がきつくないのに表現力が高い、非常に優れたものである。

そしてEasydrawの操作法はMacdrawとほぼ同じ(むしろ「Illustrator」に近い)が、ふだんMacintoshを触っている人にも違和感なく使える。操作体系を拝借するというのは、あまり頭を使っていないという意味では決してほめられた行為ではないし、このせちがらい世の中では裁判沙汰にもなりかねないが、とりあえずいまは、Macdrawの操作体系が事実上の標準、古典的な

## 「先輩」Macintoshはどうか

デバイス非依存で用紙サイズを把握した文書編集は、Macintoshの世界では遙か昔に達成されていることである。ちなみにMacintoshはこれをさらに推し進め、表示サイズと印刷サイズが同じになるようになっていく(一部機種を除く)。これは、OSのレベルでディスプレイの解像度をきちんと把握しているおかげでもある。X68000/030では、ディスプレイが15インチでも21インチでも画面のドット数が変わらず、ただドットが拡大されるだけであるが、Macintoshでは、ディスプレイを大きくすればドット数も多くなり、本当に画面が広く使えるのだ。Macintoshというのは、いろいろ気に入らない点もあるけれど、偉大な先輩であることは確かだ。

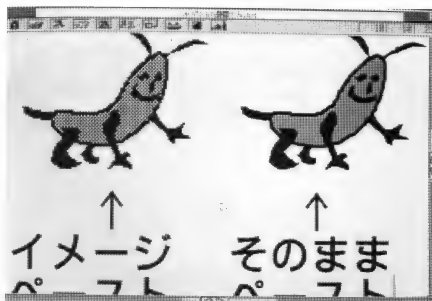


写真3 イメージペーストとそのままペースト

作法として定着していることに感謝することにしよう。

## シャーペン.Xへのペースト

8月号のレビューの内容のなかでぜひともフォローしておくこと、それはシャーペン.Xへのカット&ペーストである。8月号では、イメージペーストだけしかできないのでひどい、というようなことを書いた。

これは正確に言えば間違いである。シャーペン.Xの「そのままペースト」機能を用いることで、ドローオブジェクトを直接シャーペン.Xの文書に張りつけられる。この機能に関する記述はマニュアルではなく、README.DOCファイルに書いてあった。

ちなみに「そのままペースト」機能は、買ったときのままのシャーペン.Xでは使えない。設定のしかたは、そのREADME.DOCとシャーペン.Xのマニュアルをよく読めばわかるが、簡単に説明しておく。

- 1) シャーペン.Xを起動する
- 2) 環境メニューのキー定義書き出しを使って、キー定義を適当な名前前でセーブする
- 3) キー定義ファイルをシャーペン.Xに通常テキストファイルとして読み込む
- 4) イメージペーストを定義している部分を探す(シャーペン.Xの単語検索で「image」という文字列を探せばよい)
- 5) その行を複製する
- 6) メニューアイテム名を「イメージペースト」から「そのままペースト」に変え、image命令のオプション「-M0」を「-M-1」に変える
- 7) 修正したキー定義ファイルをセーブする
- 8) 環境メニューのキー定義読み込みを使って、キー定義ファイルを読み込む
- 9) シャーペン.Xの文書で右ボタンを押して、「そのままペースト」というアイテム

があることを確認する

これで、Easydraw.Xでカットした図形をシャーペン.Xにペーストできるようになる。

写真2をご覧ください。Easydraw.Xで描いた図形をシャーペン.Xの同じ文書にイメージペースト/そのままペーストして並べてみた(この絵は、西川善司画伯にEasydrawの使い方を説明したときに彼が即興で描いてくれたキャラクターで、憂鬱くんと呼ばれた)。

ペーストした状態では何の違いもないが、拡大表示してみると違いがわかる(写真3)。イメージペーストしたものは拡大したときにドットが粗くなって、図形がイメージデータに落ちてしまっていることがわかるが、そのままペーストしたものは拡大しても図形としての情報を保っている。

この差は印刷することによってさらに明らかになる……といたいところなのだが、シャーペン.Xでそれをいって嘘である。なぜなら、シャーペン.Xでの印刷は「イメージ印字」であり、ウィンドウに表示されているイメージをドット単位で正直に印刷するという仕様になっているからだ。したがって、ちゃんとした大きさを印刷しようとするなら、500%などといった拡大表示をしながら印刷を実行しなくてはならない(図1)。

シャーペン.Xはデバイス非依存の考え方を根本的に欠いている。ドット単位の文書編集しか考えていない。印刷するときに拡大すればいい、という考えは非常にあさはかというものである。シャーペン.Xでは、文書を、狙った大きさを描くのが難しい。A4の用紙にきっちりレイアウトされた文書を書こうとすれば、プリンタの解像度とウィンドウのドット数の関係をきちんと把握していないと無理である。文書入力ツールとしてのシャーペン.Xはいいセンといっているかもしれないが、印刷ツールとしてみればまだまだだ。

ではどんなのがいいのかといえば、つまりEasydraw.Xである。狙った大きさをきちんと図を描くことが可能になっている。ウィンドウには用紙サイズの枠があり、A4判の紙いっぱい文書をレイアウトすることも簡単である。ウィンドウにスケールもついており、「2.5cmの大きさの図形」とい



うようなものも目で見ながら描ける。プリンタの解像度をみて印刷のドット数を計算するのはプリンタドライバであり、ユーザーが面倒な計算をする必要は一切ない。

この件に関しては、シャープペン.Xはあくまでテキストエディタとして割り切り、まっとうな日本語ワードプロセッサになるはずのEG Wordの登場を待たばかりである。少なくともMacintosh版はきちんと用紙のサイズを把握した文書編集ができていた。

ともあれ、カット&ペーストがどれほど使えるものかは、Easydrawとシャープペン.Xが証明してくれることだろう。マルチウィンドウ環境のもとでこそなせる技といえる。結局はMacintoshの受け売りではあるのだが、いいものはいいのだ。

それはそうと、右ボタンでポップアップメニューが出て、カット&ペーストができるというのは、マウスのボタンが1つのMacintoshにはできない芸当だ(写真4)。まあ、結局ショートカットキーを覚えるから関係ないといえはいえる。

### 好奇心をもって使いこなそう

Easydraw SX-68Kは図形編集ソフトだから、操作の基本はマウスだが、キーボードを併用することでさらに素早く小回りの利いた操作をすることができる。

OPT.1キーによるショートカットはいうまでもない。バックスペース(BS)キーでドロオブジェクトを削除できたり、図形描画モードから抜けるためのエスケープ(ESC)キーなども覚えておくとマウスの移動量が少なくなって便利である。

それとキーボードなしだとどうしてものがフォントの種類/サイズを選択。ふつうにメニューを使うと、フォント選択ダイアログが開いて、さらにその中にあるメニューから選ばなくてはならない。操作性が異

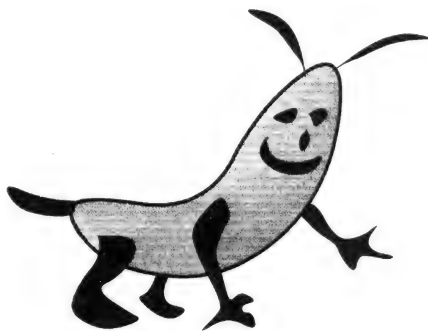
様に悪い。が、シフトキーを押しながらマウスの右ボタンを押すとフォントの種類を選ぶメニューが、コントロールキーを押しながらだとフォントのサイズを選ぶメニュー

ー(なぜかドット単位でしか選べない)が出て、素早くフォントの種類またはサイズが決められる。

しかし、私はフォントの種類とサイズの

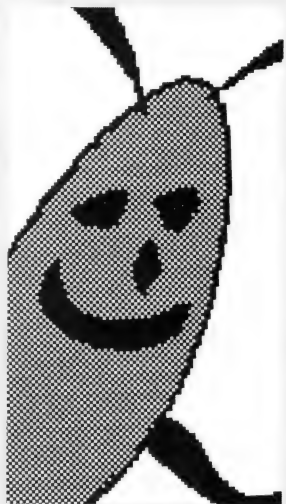
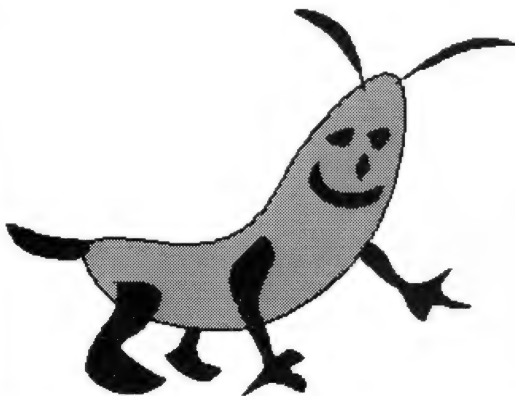
図1 Easydrawとシャープペン.Xの印字サンプル (左: 50%に縮小, 右: 原寸)

### Easydrawで描いた憂鬱くん



作…西川善司画伯

### イメージペーストした憂鬱くん



### そのままペーストした憂鬱くん

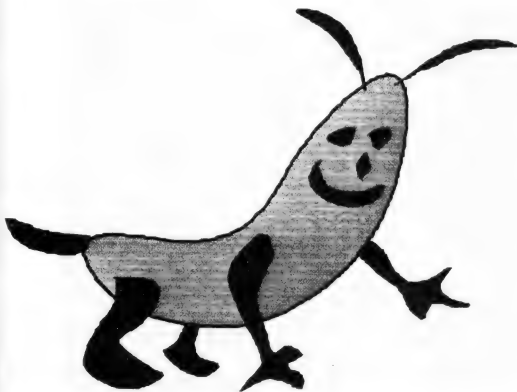


写真4 ポップアップメニューでカット&ペースト



変更を頻繁にやるので、もう少しわかりやすい操作体系にしたほうがよかった気もする。シフトキーやコントロールキー併用なんて、マニュアルを読まないと発見できなかった。せめて階層メニューで用意すればだいぶましだったとも思う。

これに限らず、マニュアルなしだときつい場面がいくつかある。が、使ってみてうっとうしいとか変だとか思う場面には、なんらかの抜け道があるので、好奇心をもっ

ていろいろつついてみるとよい。

## アウトラインフォント

すでにいわれているとおり、SX-WINDOWでは、ツアイト社から出ている和文アウトラインフォントを利用することができる。字の品質は、フォントにもよるが、なかなかのもの。当然ながらROMフォントをスミージングしたものと比べれば雲泥の

差。が、10MHzマシンだとフォントの展開が遅いので、ちとつらい。さすがにX68030(私には68882もつけてある)だとさくさくと展開してくれる。

とにかく英数字フォントの充実が望まれる。私のEasydrawの使用状況を考えると、印刷して美しい英数字というのは必須なのだ。最低限、明朝体とバランスのいいTimesと、ゴシック体とバランスのいいHelvetica。この2書体はないと話にならない。も

## イメージオブジェクトの取り扱いについて

Easydrawではイメージオブジェクトを取り扱うことができる。キャンバス、Xなどで絵を表示しておいてカットまたはコピーし、Easydraw.Xのウィンドウにペーストする。その際、ペーストする前にある設定をしておかないときれいな絵にならない。

その設定とは、「環境設定」ダイアログに収められている、「ペースト時の色変換方式」と「ペースト倍率」の設定である。

Easydrawは、カットまたはコピーした絵が65536色でも、ペースト時に白黒2階調のビットマップに変換して取り込む。このため、ペーストする前に設定が必要なのだ。

色変換方式は、コントロールパネルで出てくるものと同様のもの。デフォルトでは「ドット単位色変換」になっているため、ペーストが汚くなってしまう。「誤差分散方式」を選ぶこと。なお、コントロールパネルの色変換方式の設定をいくらいじってもEasydrawのペーストには反映されない。

倍率は、元の絵の1ドットを何ドットに変換してペーストするかという設定。1~4倍の範囲で設定できる。倍率を上げれば、中間調の表現がしやすくなっていくので印刷の画質は上がっていくが、取り込みに要するメモリと時間は増えていく。なお、倍率設定で「自動」を選んておくと、プリンタの解像度から倍率を自動的に計算する。

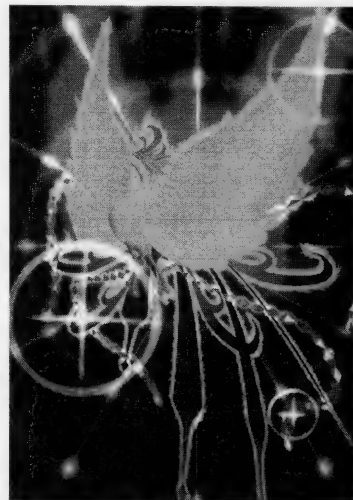
図はSX-WINDOW ver.3.0のフェニックス.PICを左から1倍、2倍、3倍でペーストしてBJ-10vで印刷したものである。

欠点もある。イメージオブジェクトを拡大縮小するとボロボロになってしまう。

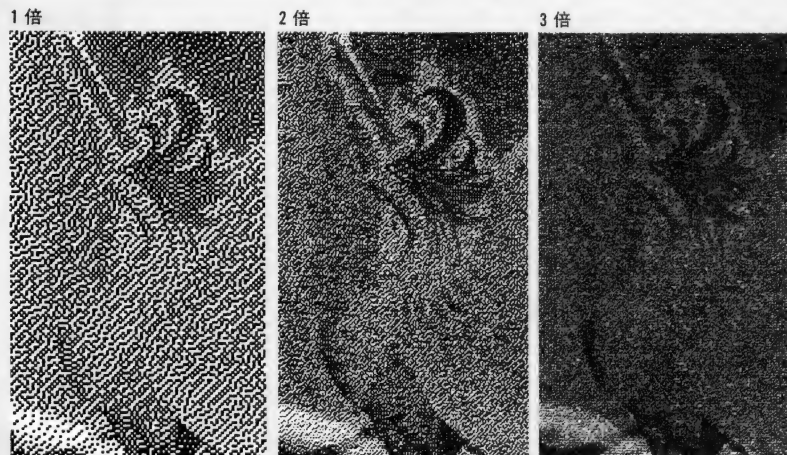
ペーストの段階でビットマップに落とすというEasydrawの実装には感心しない。イメージオブジェクトは印刷の直前までグレースケールで保持しておくべきである。参考までに、GhostScript(PostScriptを解釈して画面に表示したり非PostScriptプリンタに印刷したりできるGNUウェア)とBJ-10vで出力した例を挙げる。MacdrawとPostScriptプリンタの組み合わせでもほぼ同等の出力が得られる。画質の差は明らか。現実にGhostScriptを使えばX68000とBJ-10vでもこうした出力は実現できているから、もうソフトだけの問題なのだ。

ドローオブジェクトに貼るパターン印刷にも問題がある。プリンタによってはパターンが画面より細かくなる。煉瓦模様やハートマークなど、画面どおりに出力してほしいができない。つまりWYSIWYG的でないのだ(この点に関してだけは、PostScriptドライバはよくやっている)。逆にアメリカ風のパターンの場合は、画面のドットを忠実に再現されても困るので難しいところだが、プリンタの解像度の限界でパターン塗りつぶしをやってもつぶれてしまう。ちなみにMacdrawではこれもグレースケールで解決している。

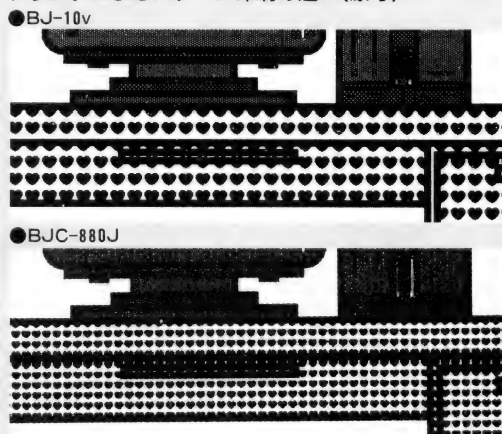
Easydrawは線画に関しては満足できるレベルに達してはいるが、トータルでみると改良の余地を残している。といってもその原因の大半は、EasydrawのほうではなくSX-WINDOWそのものにある。



GhostScriptによる出力例(参考)



プリンタによるパターン印刷の違い(原寸)





ちろん、これはEasydraw SX-68KでなくSX-WINDOWの責任だ。

## レーザープリンタドライバ

レーザープリンタドライバは、別にEasydraw SX-68K専用のドライバというわけではない。登録すればシャーペンXからでも使える。発売時期の関係からSX-WINDOW ver.3.0には含まれていないが、早いところSX-WINDOWの標準装備にするべきだろう。

さて、使い勝手であるが、プリンタによってその評価は大きく分かれることだろう。今回試したのはLIPS-IIIドライバとPostScriptドライバである。

### ・LIPS-III

キヤノンの開発したPDL(ページ記述言語)。

今回使用したプリンタはLIPS-IIIで動作するバブルジェットプリンタBJC-880J。レーザープリンタではないが、れっきとしたページプリンタであり、レーザープリンタとほぼ同様の操作感覚である(SX-WINDOWのレーザープリンタドライバは、レーザープリンタドライバというよりもページプリンタドライバといったほうがいだろう)。印字品質もなかなかのもの。Easydrawとの相性も悪くない。

### ・PostScript

アドビシステムズの開発したPDL。事実上の業界標準。Macintoshの標準レーザープリンタであるLaserWriterが搭載していることもあり、メジャー度は圧倒的である。多くのプリンタメーカーからPostScript互換プリンタが出ている。

今回使用したのは沖電気工業のMICROLINE801PS。純正の(クローンでない)PostScriptを載せているレーザープリンタである。

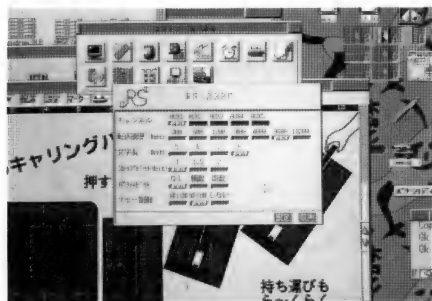


写真5 RS-232Cの通信パラメータを設定する

SX-WINDOWとの接続であるが、はっきりいうと、最悪である。あまりにひどいので、制作者がPostScriptに恨みでももっているのではないかと勘ぐってしまう。

まず、本体との接続がRS-232Cというのがいけない。ほかのドライバはLIPS-IIIにしろESC/Pageにしろちゃんとセントロニクス(X68000/030のプリンタインタフェイス)を用いているのだ。沖のMICROLINEにしても、立派にセントロニクスのインタフェイスを装備している。私の頼りない記憶では、LaserWriterはLocalTalk(MacintoshのLAN)とRS-232Cしか装備していなかったような気がするが、それにしてもあんまりというものだ。

本体とプリンタをRS-232Cのクロスケーブルで接続し、通信パラメータをコン

ロールパネルで設定する(写真5)などの面倒な手続きを経て、やっと動作する。シリアルケーブルだから、データ転送も遅い。

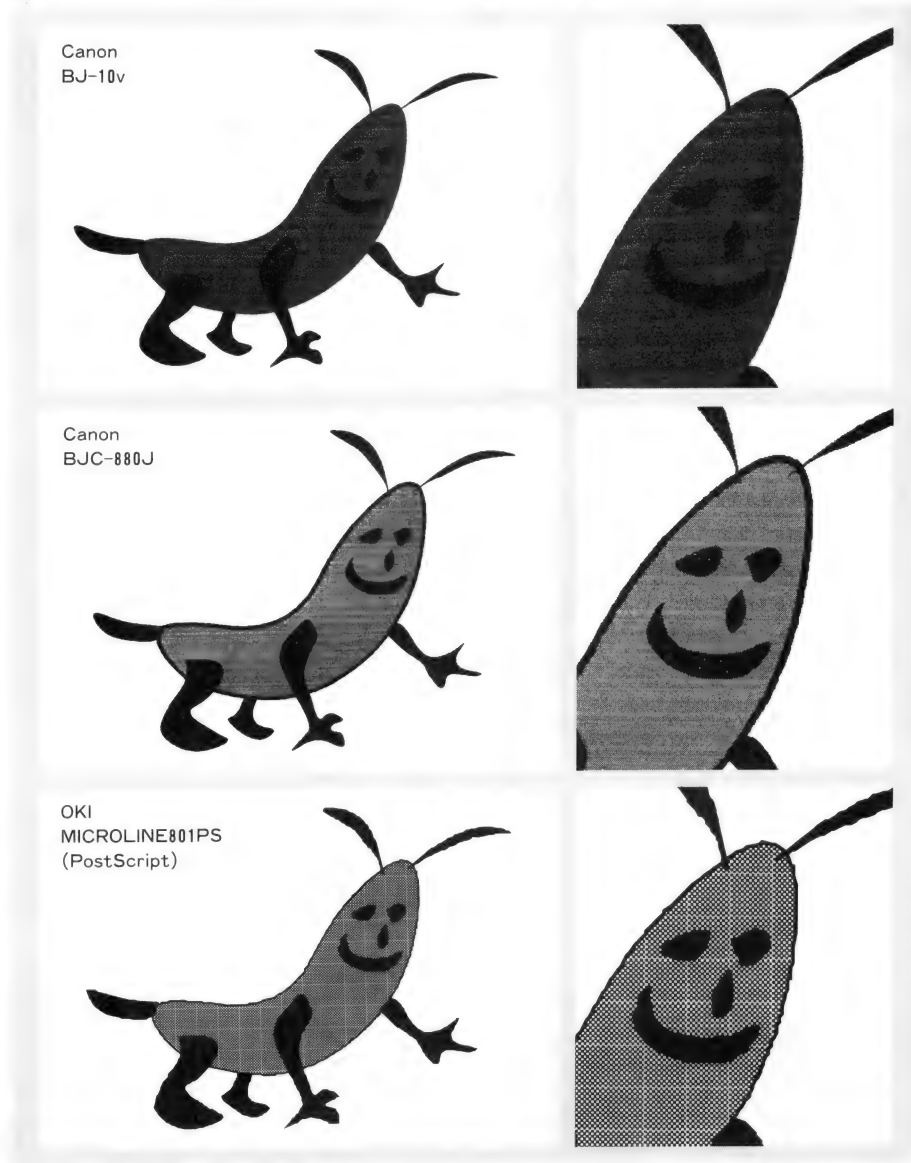
で、その苦勞に見合う成果が得られるかというと、得られない。印字品質は今回試したもののなかでは最悪である(ちなみに今回試したのはBJ-10vとBJC-880J、それにMICROLINE801PSである)。

私自身はPostScriptの崇拝者なので、どうしても納得いかない。そこで、RS-232Cケーブルの先に、プリンタでなくX68000をつないでみた。本体側で印刷を開始する直前に、プリンタのふりをしたX68000側で、

copy aux test.ps

などとやれば、PostScriptコードをテキストファイルとして取れる。それを読んでみたところ、SX-WINDOWのPostScriptド

図2 プリンタごとの印字サンプル (左: 50%に縮小, 右: 原寸)





ライバはPostScriptの機能を生かしているとは思えなかった。ベジェ曲線を線分の集合に分解してPostScriptプリンタに送るとは、まったくもっていい度胸である。美しい自由曲線を愛する者としては許しがたい。

あと、原因はよくわからないが、PostScriptドライバは根性がないのか、印刷できないものがやたらにあった。ちょっと大きな文書を印刷しようとする、とたんに長々と黙り込んだりエラーを出したりする。

これらはレーザープリンタドライバの責任であり、Easydraw SX-68KやMICRO LINE801PSの責任ではない。とにかく、現時点ではPostScriptドライバは使いものにならないのだ。

あまり執着していると、Macintoshのよさばかりが目立ってしまうので、現時点では無視するのが得策といえる。

## 結局はお買い得

一部には問題があるが、とにかくお買い得のソフトだといっておく。私はこのソフトを使うようになってから、SX-WINDOWを立ち上げる時間が増えた。これでEG Wordが出れば、多くのユーザーがSX-WINDOWに移行すると確信する。長年、SX-WINDOWはおまけでしかなかった。パワーユーザーのなかには、SX-WINDOWなんて使わないという意味のことを得意げにいう人も少なからずいるようだが、そろそろ振り向いてもよからうと思う。

同程度のスペックのMacdrawに比べて、遙かに安いというのはすごいことだ。MacdrawとEasydrawの価格差は、定価ベースでも3倍じゃきかない。まあ、あれだけ売れているのに、いっこうに値を下げないクラリス/システムソフトも悪いといえは悪いのだが……。



X68000用  
シャープ  
3.5/5"2HD版 19,800円(税別)  
☎03(3260)1161

## アウトラインフォント

本文中で述べたとおり、SX-WINDOWでは、ツァイト社から出ている和文アウトラインフォントを利用することができる。ここで整理しておく。とりあえず販売形態によって分類する。

### 1) 書体倶楽部フォント

ツァイト社のアプリケーションから利用できるフォントをアプリケーションから独立して販売しているもの。Z'sSTAFF PRO-68Kで利用できることでお馴染みであろう。

アウトラインの構成要素は線分。拡大すると輪郭が多角形になっているのがわかる。

### 2) JGフォント

同社のワードプロセッサ「Z'sWORD JG」用のアウトラインフォントで、WINDOWS対応ドライバも出ている。

アウトラインの構成要素はベジェ曲線。拡大しても曲線の輪郭を保てる。

### 3) アプリケーション添付フォント

たとえばZ'sSTAFF PRO-68K(ver.2.0以降)には、書体倶楽部フォントのサブセットが入っている。

\* \* \*

書体倶楽部フォントとJGフォントは、基本的に同じフォントデザインでデータ構造を変えたものである。JGフォントのラインアップを見てみよう。なお、JGフォントシリーズの販売はアスキーが行っている。

#### ・基本フォントセット

明朝体、角ゴシック体、丸ゴシック体の3書体。WINDOWS対応フォントドライバ(18,000円、税別)に付属している。

\* \* \*

#### JGフォント 基本フォントセットの明朝体

その微妙にして優美なシルエットを愛するようになれば、あなたもマニアの仲間入りといっていいいでしょう。すらりと大胆に伸びる曲線に新鮮な感動をおぼえ、ほんのちょっとしたくねりに魅了される……この快感、ああくせになりそう。でも新たな歓びを得るためには投資が必要なのでした。

#### ・明朝セット(28,000円、税別)

JTCウィン明朝体細字、JTCウィン明朝体太字の2書体。

#### ・角ゴシックセット(38,000円、税別)

JTCウィン角ゴシック体細字、JTCウィン角ゴシック体中太字、JTCウィン角ゴシック体太字の3書体。

#### ・丸ゴシックセット(38,000円、税別)

JTCウィン丸ゴシック体細字、JTCウィン丸ゴシック体中太字、JTCウィン丸ゴシック体太字の3書体。

#### ・タイトルセット(38,000円、税別)

JTCウィン極太明朝体、JTCウィン極太角ゴシック体、JTCウィン極太丸ゴシック体の3書体。見出しなどに用いる。

#### ・応用セット(28,000円、税別)

毛筆体、教科書体の2書体。

\* \* \*

利用頻度の高いのは、なんといっても明朝体と角ゴシック体である。文字はただ読めればいいというものではない。フォントの骨格や輪郭の微妙な曲線にまでこだわらなくてはDTPを語る資格などない。そこで明朝体と角ゴシック体について、私の独断による寸評を加えてみたい。

\* \* \*

#### ・基本フォントセット

書体倶楽部では「新明朝体」「新ゴシック体」と呼ばれているフォント。新明朝体と新ゴシック体はZ'sSTAFF PRO-68K ver.3.0にも付属している。

フォントデザインはほかに比べるとかなり見

#### Z'sSTAFF PRO-68K ver.2.0の明朝体

その微妙にして優美なシルエットを愛するようになれば、あなたもマニアの仲間入りといっていいいでしょう。すらりと大胆に伸びる曲線に新鮮な感動をおぼえ、ほんのちょっとしたくねりに魅了される……この快感、ああくせになりそう。でも新たな歓びを得るためには投資が必要なのでした。

曲線に  
え、ほ

曲線に  
え、ほ



劣りする。骨格が弱々しいというのか、読んでいて頼りない。おまけフォントの色彩が強いのだろう。

#### ・明朝セット、角ゴシックセット

「JTCウィン〜」と名のついたフォントは、日本情報科学(株)で開発されたフォントで、それなりにしっかりしたフォントデザインをしている。書体倶楽部でも同等品がある。値段も少々高めである(といっても内容を考えればお得である)。

フォントデザインは特徴的。悪くいえば多少くせがある。個人的には嫌いなデザインではないが、好き嫌いが分かれるかもしれない。

通常の用途ならば「〜明朝体細字」と「〜角ゴシック体細字」「〜角ゴシック体中太字」があれば十分。どうして明朝体2書体、角ゴシック体3書体をそれぞれセットにするか疑問ではある。

#### ・昔の明朝体、ゴシック体

現在製品としては存在しないが、その昔、Z's STAFF PRO-68K ver.2.0に添付されたフォントというものがあつたのだ。今回ふと思い出して、引っぱり出してみたが、意外にフォントデザインがいいのだ。第1水準しかないのが惜しい。

デザインの悪いフォントは、長い文書の印刷に用いたときは特に読んでいて疲れるものだが、この昔のフォントは、骨格が素直で、今回紹介したもののなかではいちばん読みやすい。

書体倶楽部形式(多角形)だし、輪郭の品質も決まっているとはいえないが、それでも注目に値する。どうしてこのフォントをきちんと育てないで新明朝体や新ゴシック体に走ってしまったのか、理解に苦しむところである。

#### JGフォント 明朝セットの明朝体細字

その微妙にして優美なシルエットを愛するようになれば、あなたもマニアの仲間入りとっていいでしょう。すらりと大胆に伸びる曲線に新鮮な感動をおぼえ、ほんのちょっとしたくねりに魅了される……この快感、ああくせになりそう。でも新たな歓びを得るためには投資が必要なのでした。

曲線に  
え、ほ

\* \* \*  
結論としては、ショップに買いにいくならJGフォントの明朝セットと角ゴシックセット、もし入手経路があればZ'sSTAFF PRO-68K ver.2.0のフォントがお勧めということになる。

#### JGフォント 角ゴシック中太字

その微妙にして優美なシルエットを愛するようになれば、あなたもマニアの仲間入りとっていいでしょう。すらりと大胆に伸びる曲線に新鮮な感動をおぼえ、ほんのちょっとしたくねりに魅了される……この快感、ああくせになりそう。でも新たな歓びを得るためには投資が必要なのでした。

曲線に  
え、ほ

#### JGフォント 基本フォントセットの角ゴシック体

その微妙にして優美なシルエットを愛するようになれば、あなたもマニアの仲間入りとっていいでしょう。すらりと大胆に伸びる曲線に新鮮な感動をおぼえ、ほんのちょっとしたくねりに魅了される……この快感、ああくせになりそう。でも新たな歓びを得るためには投資が必要なのでした。

曲線に  
え、ほ

推測だが、Z'sWORD JGの昔のバージョンには、Z'sSTAFF PRO-68K ver.2.0と同じデザインのフォントが第2水準まで揃っているのではないかとらんでいる。が、いまではもう売っているはずもなく、どうしようもない。

#### Z'sSTAFF PRO-68K ver.2.0のゴシック体

その微妙にして優美なシルエットを愛するようになれば、あなたもマニアの仲間入りとっていいでしょう。すらりと大胆に伸びる曲線に新鮮な感動をおぼえ、ほんのちょっとしたくねりに魅了される……この快感、ああくせになりそう。でも新たな歓びを得るためには投資が必要なのでした。

曲線に  
え、ほ

#### ROM24ドット+スムージング

その微妙にして優美なシルエットを愛するようになれば、あなたもマニアの仲間入りとっていいでしょう。すらりと大胆に伸びる曲線に新鮮な感動をおぼえ、ほんのちょっとしたくねりに魅了される……この快感、ああくせになりそう。でも新たな歓びを得るためには投資が必要なのでした。

曲線に  
え、ほ

すべてBJ-10vによる印字例(上:20%に縮小,下:原寸)



# 3.5インチFDDを改造する

満開製作所 Nakamura Takao 中村 隆生

X68000 Compact XVIの内蔵ドライブを2DD対応に改造しましょう。

なんとこれが意外と簡単な改造ですんでしまいます。

なお、実際に作業を行う場合は各自の責任で行ってください。

CZ-674C (Compact XVI)のFDDは2DDディスクを読み書きすることができません。CZ-674Cを改造してRED ZONEとして売っている満開製作所としては、売り文句を増やすため2DD対応をすべく考慮を迫られるところで、一時はドライブ換装サービスまで企画されました。んが、ドライブに簡単な改造を施すことで2DDに対応できるということがわかりましたので、報告します。筆者の改造実績はまだ4基だけですが、どれも正常に動作しています (たぶん)。

## 2DDを読む

2DDが使えるようになるということでのようなメリットが生まれるのでしょうか？

3.5インチ2DD (720Kバイト) MS-DOSフォーマットは世界的な標準ディスクフォーマットです。世界で唯一の、3.5インチ2DDが読めないマシンといわれたX68000 Compactもこれで世間並みの互換性が取れるようになります。必然的に、仕事や趣味でAT互換機を使っている人は、フロッピーで直接データのやり取りができるようになります。それから、PC-9801用のソフトでごくまれに2DDで販売しているものがありますが、こういったディスクも直接読めるようになります。つまり、他機種とのデータのやり取りが楽になるということです。

余談ですが、Macintoshの昔のタイプのディスクやAMIGAフォーマットのディスクは読めませんでした。フォーマットが違うんですね (でも2DD MS-DOSフォーマットも使えるから大丈夫)。まあ、ハードウェアの制約からってMacintoshは無理としても、ソフトウェア次第ではAMIGAとかポータブルワープロ機などの特殊なフォーマットのディスクを直接読むこともできないはないでしょう。

さて、これまでできなかったものがどうしてできるようになるのでしょうか。

結論からいってしまえば、実はCZ-300C (X68030 Compact)のFDDとCZ-674CのFDDは、基本的には同じものなのです。CZ-300CのFDDにはメーカーの手によって2DD対応がなされているのですが、その改造を自前でやっちゃおうというわけですね。具体的には、FDDユニットの中に組み込まれている制御基板に、1カ所ショートジャンパを飛ばすだけなのです。

しかし、てりめえではありますが、FDDの分解の前に、本体の分解をしなくてはなりませんね。過去にCompactタイプの分解法に触れたことはありませんから、まず本体の分解手順を解説しましょう。

ほかの用途にも使えるので、以下に挙げるものは揃えておいて損はありません。

●でかいドライバー。もちろんプラスとマイナスが必要です

●精密ドライバー。これもひと揃いあったほうがよろしい

●ラジオペンチとニッパー。必需品ですね

●ハンダごて。こて先クリーナがあるとモアベターよ

●すずめっき線か細いワイヤー。ジャンパ線です

●メンディングテープ。基板の上にメモしたり、コネクタをまとめたり、コネクタをどこかへ仮止めしたりするとき非常に便利です

●軍手。シールドで手を切らないための対策で、あれば安全という程度です

## 本体を分解する

相手は電子部品ですから、分解の前には、ドアノブや水道の蛇口などに触りまくって体の静電気を逃しておくことが望ましいというのはいまでもありません。

1) まず本体底面のビスをはずします。底

面は脚になってますが、その脚を展開するための4本のビスははずさなくていいです。背面の方向にあるビスを1個はずしてください。それから、まるごと前面方向に水平にずらして持ち上げると、この脚がはずれます (写真1)。

2) フロントパネルをはずします。若干厄介ですから気をつけて。シャーシに対して4カ所ぐらいのツメがかかっています。心持ちなめ方向にひねるようにすると楽にはずれます。そのはずです。文章での解説は面倒ですね (写真2)。

3) 側板は、本体上面と背面のところにツメがかかっていますが、その前に底面でリアパネルとツメがかみ合っています。ですから、底面のツメをはずすように垂直に持ち上げたあと (このとき側板はしなりますが、別に気にしなくてよいです)、上面と背面のツメを水平方向にはずします。

4) リアパネルは、まず底面のツメをはずします。背面は2カ所ツメがかかっていますが、これは後ろの方向に浮かせるとはずれます。最後に、上面がツメ2カ所で留まっています。今回はリアパネルと次の上面パネルは無理にはずす必要はありませんが、参考までに。

5) 本体上面のパネルは、だいたい中央の位置にある、コの字型のツメでひっかかっています。このツメはまともにはずそうとせずに、パネルの両端を浮かせてから全体を回転させると、楽にはずれます。割と丈夫な部品ですから安心してください。

6) これで、外面のパネルはすべてはずれて、シールドだけの状態になりました。シールドは2枚構成で、うち片方は下のほうが黒い絶縁テープで留めてあると思います。はがしてください。あとはシャーシに乗っかっているだけです。簡単に取れます。もう一方の、シャーシを包むようにして装着してあるシールドは、とりあえず今回ははずす必要はありません。くれぐれもシー



ルドで手を切らないように。とても痛いですが。

この時点でFDDユニットが出現してま  
すから、本体の分解はここまででよいです  
(写真3)。が、メイン基板を見たいという  
場合には、さらにI/Oスロット、電源ユニ  
ット、フロント基板ユニット、スピーカーコ  
ネクタをはずし、残りのシールドをはずす  
必要があります。しかし、そこまでやると  
今回の主旨からははずれますから、また別  
の機会に譲ることにしましょう。

7) FDDとメイン基板との間は、白いフラ  
ットケーブルで2カ所配線してありますが、  
これは引き抜くだけではずれます。あとで  
組み立てるときのことを考えると、FDD側  
をひっこ抜いたほうがよいでしょう。ケー  
ブルをはずしたら、いよいよFDDユニ  
ットをはずします。ユニットはツメ4カ所(片  
側2カ所ずつ)で留まっています。片側ず  
つ持ち上げると楽でしょう。アレでしたら  
友達か誰かに手伝ってもらうのも手です  
ね。これでFDDユニットの取り出しは完了  
です(写真4)。

しかし、最初に分解したときは、このパ  
ズルのような組み立て方には驚かされま  
した。ここまでではずしたビスは、実に1本  
だけです。モックアップを作るのは、さぞ  
かし大変だったことでせう。

## ユニットを分解する

ユニットを取りはずしたら、ようやくこ  
れを分解にかかれます。精密ドライバーと  
ラジオペンチがないと難儀するので、用意  
しておきましょう。コネクタ類は用心のた  
め、すべてラジペンを利用してひっこ抜  
きます。

1) 左右2枚のスペーサーが、2基のドラ  
イブをはさみ込むようにして、8本のビス  
で留まっています。そのビスをはずせば、  
ドライブが単体にバラけます(写真5)。こ  
のあとの作業は、ドライブごとに1回ずつ  
2回行うわけですが、片側のドライブさえ  
2DD対応になればいいというのであれば話  
は別です。

2) ドライブ本体の片面には、まるまるシ  
ールドが被さっています。それをはずすわ  
けですが、基板を取り出すまでは、この方  
向を上面として解説します。このシールド  
は4カ所のツメでひっかかっていますが、

マイナスの精密ドライバーでこじ開けると  
よいでしょう。

3) シールドをはずすと、緑色をした基板  
が見えますね(写真6)。この基板が改造の  
目標ですから、これを取りはずします。以  
後の説明文は、ディスクドライブがディス  
ク挿入口を向こうのほうに向けて置いてあ  
るものとします。基板の手前のほう、中央  
よりやや右に、ビスが1個ありますからは

ずしてください。これは精密ドライバーで  
ないと回りません。ネジ止め剤が使ってあ  
るらしく、ちょっときつめです。

4) お次はケーブル類をはずします。合計  
5カ所・7本です。いずれも、基板と水平  
方向にひっこ抜くだけではずれます。フラ  
ットケーブルを引き抜くときのコツですが、  
ケーブルの端っことはプラ板のようなもの  
で補強されていますから、そこをラジペンでつ

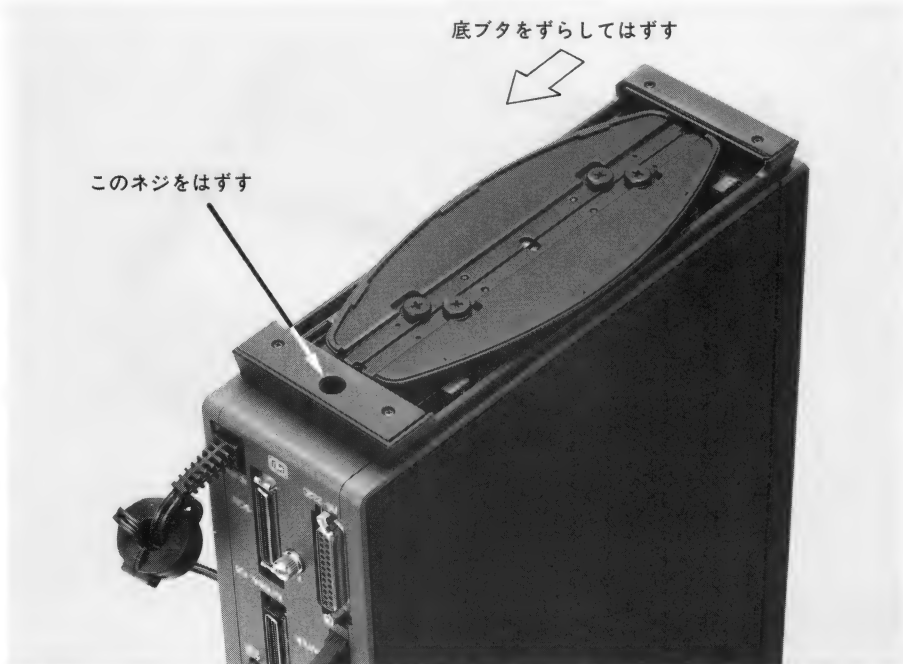


写真1

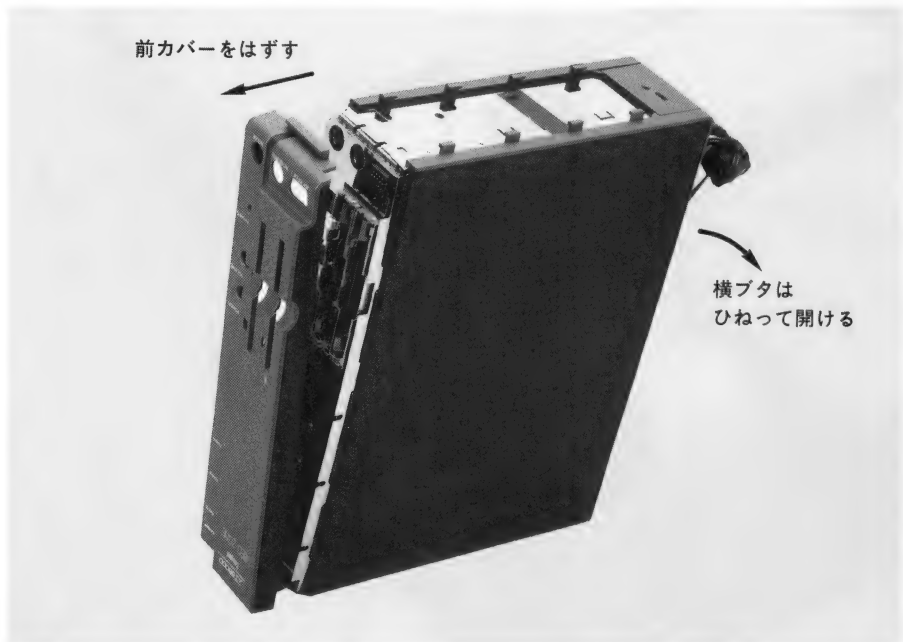


写真2

まんで引っ張ってください。コネクタは、プラの部分をラジペンでつまんで引っ張ります。間違ってもワイヤーを引っ張らないように。折ったり切ったりしてしまうと、全交換という憂き目が待ってます。というのは脅して、満開製作所に部品のストックが若干ありますから (RED ZONEのドライブ初期故障で発生したものです)、もしやっ

ちゃったら相談してください。

取りはずすケーブル、コネクタを、基板の向こう側から順に解説すると (写真6参照)、

④イジェクトボタンにつながっている、3線の黒いコネクタ。

⑤スピンドルモーター基板 (奥のほうに見える別の基板) につながっている、白いフ

ラットケーブル。

④ヘッドにつながっている、びろ〜んと長い、茶色っぽい半透明のフラットケーブル。これは白いテープで留めてあって、しかもケーブルの先が二股に分かれてますが、当然ながら両方ともひっこ抜いてください。

⑤ヘッドユニットの左にある、4線コネクタが2段についている黒いコネクタ。これは奥まったところにあって、最初よくわからないかもしれませんが。上のほうのコネクタは0トラックセンサー (だと思う) に、下のほうはオートイジェクト用のアクチュエータ (だと思う) につながってます。

⑥いちばん手前の、ステッピングモーターにつながっている、茶色っぽい半透明のフラットケーブル。このケーブルだけは (見ればわかりますが) 基板と垂直方向にひっこ抜きます。

5) ケーブルをすべてはずし終わると、基板がはずれます。その前に、透明なシールドといっしょに、細い鉄の棒状の部品と基板とがかみ合っています。この棒状の部品、はずしてしまうと初めのうちは方向がわからなくなるかもしれませんね。スケッチを残しておくか、もう片方のドライブを参考にするかしてください (写真7)。

6) 基板がはずれたら、これ以上分解する必要はありません。筆者は、同型のドライブを完膚なきまで分解しましたが、二度と組み立てることができなくなったのはいうまでもありません。

さて、基板の部品面のほうを見てください。基板の方向はいままでどおりとしますね。ふよふよしたスポンジが上のほうに2カ所ついていますが、目標の工作箇所はこの下に隠れています。というわけで撤去する必要があります。あとで組み立てるときにまた貼りつけないといけませんから、注意深くはがしてくださいね。はがすのは、LSIに被さっているほうだけでいいです。

具体的なスポンジのはがし方ですが、LSIの足には灰色の接着剤が盛ってあって、その部分とスポンジの間に隙間ができています。そこへ精密ドライバーのいちばん小さいマイナスなどを突っ込んで、スポンジを持ち上げてやりましょう。基板に傷をつけないようにね。スポンジの上のほうを引っ張ると、ちぎれちゃいます。

7) スポンジがはがれたら、LSI (MB8855)

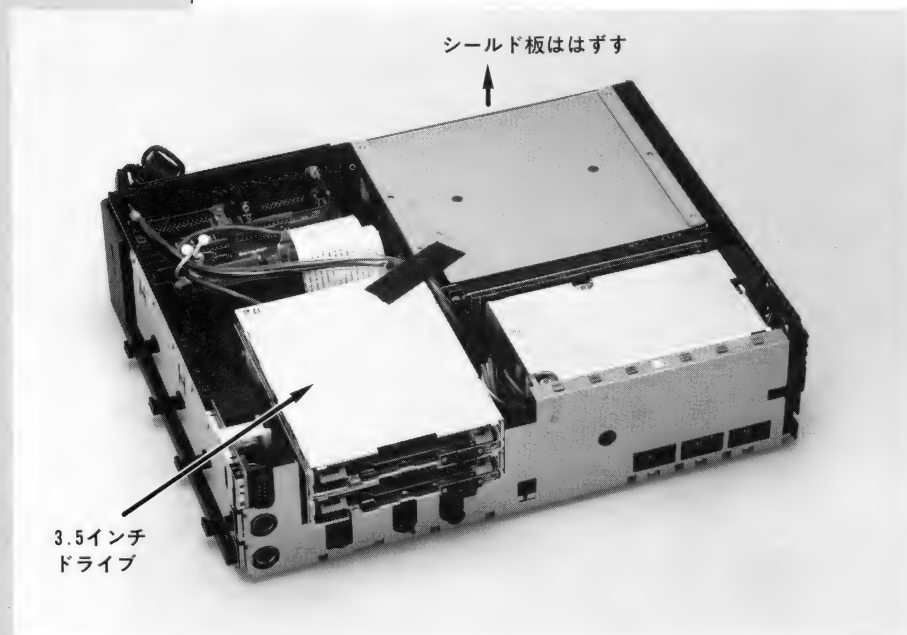


写真3

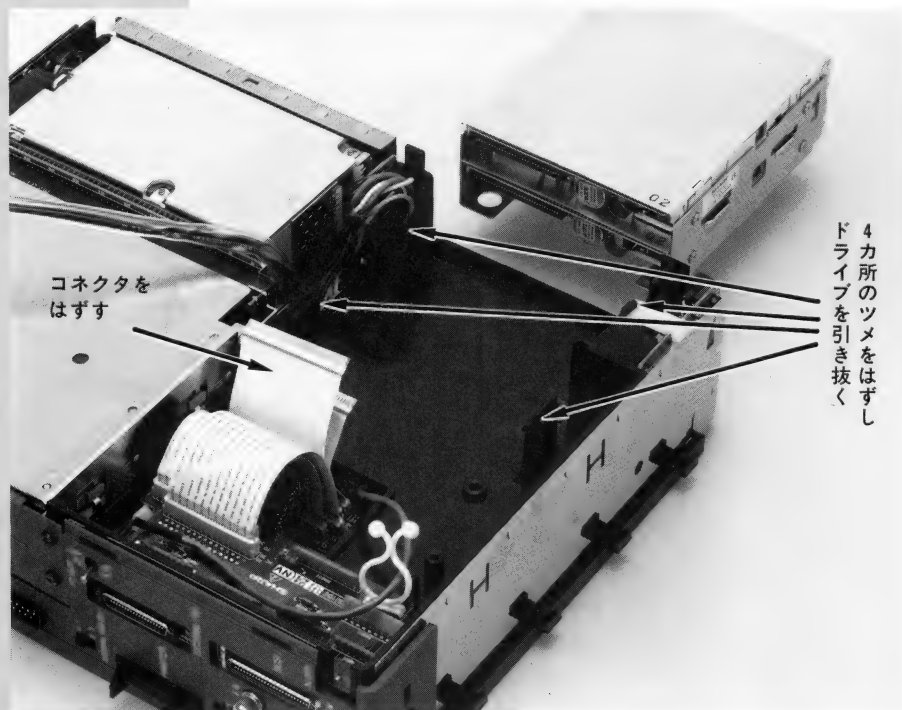


写真4



の上を見てください。JJ2と書いてあって、ハンダが四角い点のように2カ所盛ってありますね。ここが、ジャンパを飛ばすべき箇所です(写真8)。

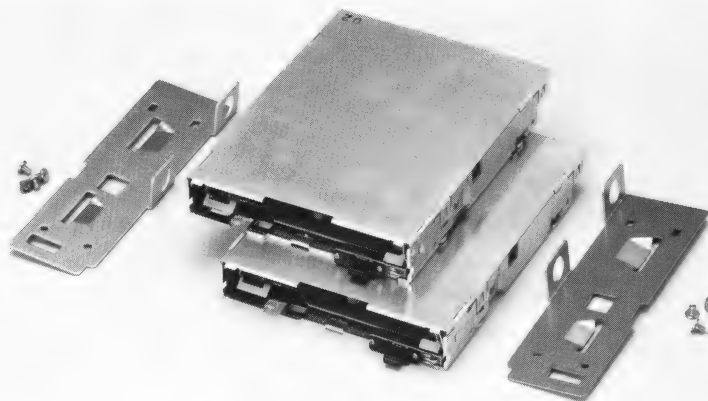
で、肝心のジャンパですが、普通のワイヤーを短く切ったもので代用してかまいません(本来は0Ωのチップ抵抗を使うべきなんだそうですが、入手が面倒です)。要は、電流が抵抗なく流れたらそれでいいんです。ただ、小さい箇所だけに、こういう加工に慣れていない人は難儀するかもしれませんね。頑張ってください。切ったワイヤーを基板の上に落とさないように注意。LSIの足の間に入り込んだりしたら悲惨です。あと、隣にあるスルーホールにハンダがかからないようにしてください。やけどには気をつけてね。

8) 加工が終わったら、元の通りに組み立てます。スポンジを貼り付けるのを忘れないように。次にケーブルを接続しますが、まず2段になっている黒いコネクタから、しかも上のほうから組み立てたほうがスムーズに作業できるようです。どのコネクタもさかさまに差せてしまいますから、方向には注意してください。不安なときはもう片方のドライブを参考にしてください。

9) あとは、分解した順序と逆に組み立てればよいのですが、3つほど注意点を。

(a) ドライブをユニットに組むときのコツですが、まず2基をくっつけて、イジェクトスイッチが上になるように縦に立てます。そのときに、コネクタ側から向かって右が0ドライブ、左が1ドライブになるのが正しいです。ドライブ番号が0か1かを見分けるには、ステッピングモーター左隣のジャンパスイッチで見ればいいです。JJ1-0が0側になっていればドライブ0、1側ならドライブ1です。

(b) パネルの取り付けは、取りはずしと同様、多少コツがいります。サイドパネルの取り付けは、まず上面と背面のツメを合わせるようにしてから行います。フロントパネルの取り付けの際は、パネルの底面のほうとシールドとがぶつからないように注意してください。前面方向に突き出しているシールドとも、ツメ状の突起でかみ合うようになっています。フロントパネルがちゃんとハマらない場合は、たぶんパネルがボリュームかPHONE端子に引っかかっているのが原因です。どうしてもうまくはま



両横のネジ（4つずつ）をはずす

写真5

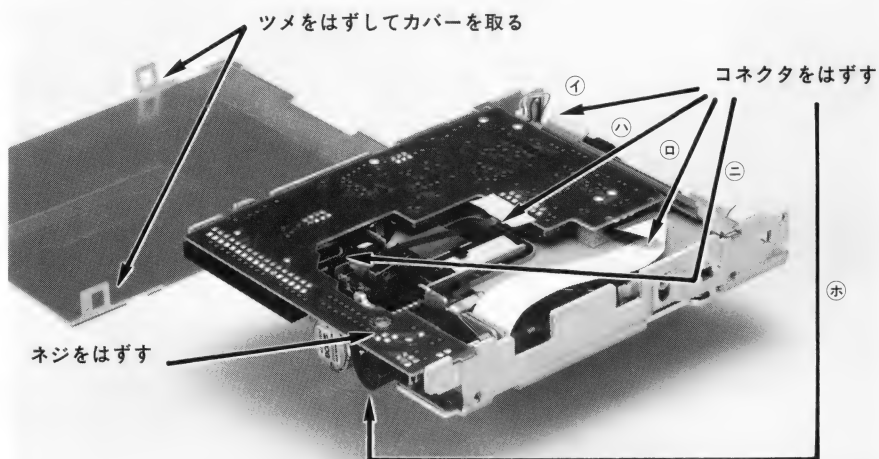


写真6

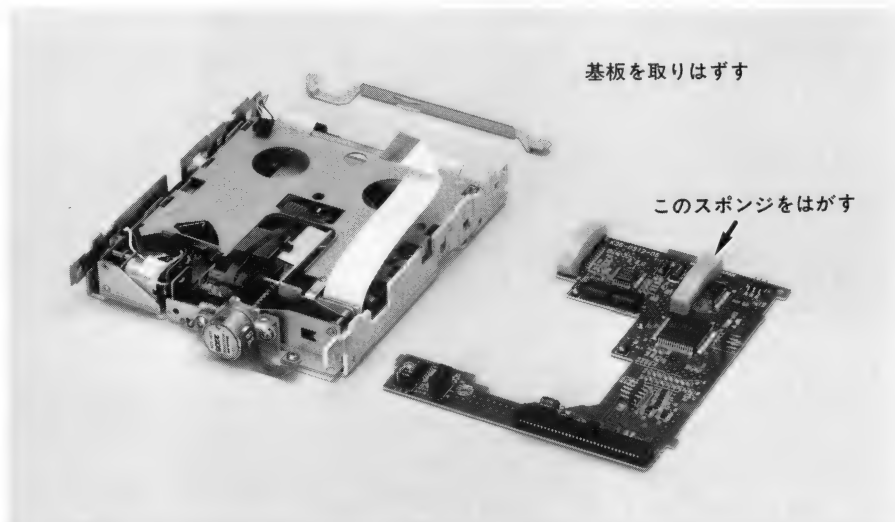


写真7

らない場合は、PHONE端子にドライバー  
かなんかを突っ込んで調整してください。

(c) いちばん最後に組み立てることになる、本体底部の脚ですが、これをネジ止めするときにネジが若干斜めになります。が、気にしないでねじ込んでください。そのうちまっすぐになります。

組み立てが終わったら、ドライブ0・1とも、まずFDを入れてみて、イジェクトランプが正常に点灯するか確認してください。次に、イジェクトしたり、ディレクトリをとったりするなど、ひととおり動作試験をしてみてください。正常に動作しない、コゲ臭い、異音がするなど、なにかおかしいなと思ったら、すぐに電源を切ってください。原因はおおかたコネクタの接続ミスとあったところでしょう、再度分解して確かめてください。

## 実際に使ってみる

説明するまでもないと思いますが、Human68kのver. 3.0から付属しているFDD EVICE.Xを使えば、640K/720Kバイトの2D Dディスクを読み書きすることができます。ですから、ソフト面では(Human68kver.

3.0さえ持っていれば)まったく問題ないわけ  
です。Human68kのver.3.0はSX-WIN  
DOWのver.3.0についてます。持っていない  
人は買ってください。

ところで、私の記憶が正しければ、ディスクを2DDでフォーマットする方法はまだ公表されていなかったように思います。実は、ver.3.0付属のFORMAT.Xから、隠し機能として2DD、2HCなどのフォーマットができるようになってます。ただし、メニュー画面からは設定できないので注意。

指定はコマンドラインからFORMATに続く次のようなオプションパラメータで行います。

/4: 2HD(IBM 1.44Mバイト)

$$/5 : 2\text{HC}$$

/8: 2DD(640Kバイト)

/9 : 2DD(720Kバイト)

例) FORMAT B: /9

数字がてたらめのように見えるかもしれませんが、ちゃんと根拠があってですね、4は1.44の4、5は512バイト15セクタの5、8は8セクタ、9は9セクタという具合になってます。確かMS-DOSのFORMATコマンドも、似たようなスイッチ体系だったと思います。

ところで1.44Mバイトというやつですが、

FDDEVICE自体が対応していないためか、フォーマットできるように見えて実は失敗します。注意してください。フリーウェアかなんかで9セクタドライバがあれば、成功するかもしれません。試したことがないのでよくわかりませんが、もしかしたらドライブ自体に問題があるのかもしれません。

## 改造のススメ

こういう改造の場合のお約束ごとですが、  
 改造作業はあなたの責任で行ってください。  
 失敗したからといってOh!Xや満開製作所  
 に文句をいわれても困ります。が、改造自  
 体は簡単なので、Compactをお持ちの方は  
 挑戦されてはいかがでしょうか。

自分で改造するのはやだという方のために、満開製作所でも改造サービスを行います。ただ、あまり引き受けたくないの(だって人手が足りないもの)、送料込みで税別1万円という人をナメきった価格設定になっています。文句なら社長にいらしてください。この件のお問い合わせは、

〒171 東京都豊島区長崎1-28-23

Muse西池袋2F 株式会社満開製作所

**☎03 (3554) 9282**

までお願いします。

参考までに、RED ZONEなどは業者さんを使って改造してますが、この2DD改造作業を行うのは私ということになってます。冗談キツイとは思いますが<sup>3</sup>、社長のお達しなのでしかたありません（改造したドライブのシールドの裏にサインしようかな）。

なぜこの改造方法がわかったのかは明かしをしましょう。FDDのメーカーさん(ある約束があって名前は明かせません)から、ドライブの仕様書をいただいたからです。つまらないですね。

最後に、この記事を書くにあたって、シャープのAVCさん、シャープエレクトロニクス販売西東京さん、FDDメーカーさんからは、資料を提供していただきました。いつもお世話になっております。中村ちやぶに氏からは、示唆に富んだアドバイスをいただきました。ありがとうございました。

### 參考資料

- 1) CMF0018CEZZ承認用仕様書
- 2) CZ-674Cサービスマニュアル（シャープ電子機器事業本部商品信頼性管理センター）

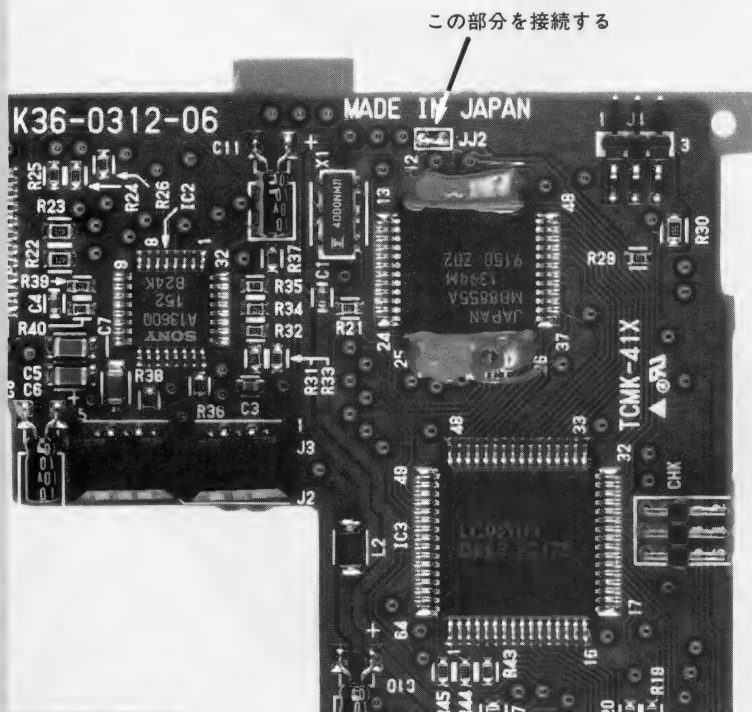
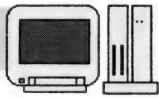
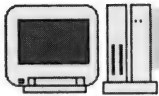


写真8





# 電話回線を使った転送アプローチ

電機本舗 由井 清人 Yui Kiyoto

今回は一般の電話回線を使ってデータ転送を行うことを考えてみましょう。まずは簡単な通信ソフトを作ってみましょう。同時に電話回線で通信する際に必要となるモデムについての基礎知識も解説していきます。

前回まででRS-232Cを利用しX68000同士、またDOSマシンとの最低限度の接続を実験してみました。これでRS-232Cの基礎的な使い方はわかったことと思います。

さて、今回は趣向を変えまして、電話を介してのファイル転送にチャレンジしてみます。

この転送はモデムを利用しRS-232Cの信号を電話回線に接続する方法です。友人間、また仕事などでパソコン通信のBBSを介さず直接転送したいという要求は多いのではないのでしょうか。

まず、この方法には少なくとも2つの場合が考えられます。

- 1) パソコン通信ソフトを双方で利用する
- 2) 独自にモデムを制御し転送する

ここでは、独自にモデムを制御する方法を重点的に紹介します。まず市販ないしフリーソフトウェアなどのパソコン通信ソフトによるアプローチを簡単に説明します。

## ●モデムとはどういうものか

すでに多くの皆さんはモデムを所有していると思います。また、モデムがどういうものであるかは漠然と理解しておられるでしょう。

おさらいとして説明するならば、次の機能を備えた機材といえます。

- 1) 相手に電話をかける機能がある

といえば簡単ですがこれをNCU(Network Control Unit)、直訳で網制御装置といいます。要は、交換器にたくさんぶらさがった相手を選択する機能、つまりダイヤルできることですね。

- 2) 電話がかかってきたら受ける機能がある

普通は使いませんが、BBSホストなどは受け専門です。また、今回はこの機能を使用します。

図1 電話回線を使ったファイル転送



- 3) データ伝送機能

RS-232Cのデータを電話回線に流し込みます。また、電話回線から流れ込んできたデータをRS-232Cへ入れる機能です。

さて、これらの動きがどのようなものかは流れ図にしたほうがわかりやすいかもしれません。パソコン通信でアクセスするさまを図2のようにします。

## モデムの使い方

モデムは通常、パソコンより命令を発行して制御されます。命令はRS-232Cより出します。この命令はヘイズATコマンドと呼ばれています。これはもともと米ヘイズ社のモデムで使用していたコマンドで、命令がすべて“AT”で始まるのでこの呼称がつけました。

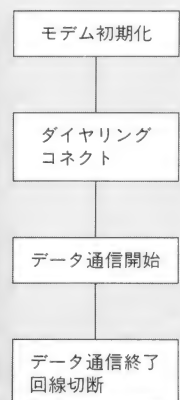
モデムにはコマンドモードとデータモードの2つがあります。

コマンドモードとは“AT”コマンドを受け付けてくれる状態です。通常、モデムの電源を入れたとき、また、アクセスするときはこの状態です。

データモードとは相手とデータのやり取りをしているときの状態です。このとき、X68000からRS-232Cを介して入ってくるデータはすべて電話回線へ(データとして)送信されます。また、電話回線から入ってくるデータは無条件にRS-232CよりX68000に送られるのです。 図2

## 20行で作る通信ソフト

さて、実際にモデムを動かしてみましょう。  
リスト1に簡単なパソコン通信ソフトETERM



```

1: #include <stdio.h>
2: #include <ioclib.h>
3:
4: void main()
5: {
6:     int c;
7:     printf( "終了はESCキーを押す\n" );
8:     while( 1 ) {
9:         if( LOF232C() ) {          /* 受信文字はあるか? */
10:            c = INP232C();           /* Yes.& 読み取り */
11:            printf( "%c", c );      /* 表示 */
12:        }
13:        else if( (c=INP232C()) ) { /* キー入力チェック */
14:            if( c==0x1b ) {          /* ESC入力終了? */
15:                break;
16:            }
17:            OUT232C( c );           /* 入力ありモデムへ出力 */
18:        }
19:    }
20: }

```

を作ってみました。わずか20行です。最低限の機能しかありませんが、これで、真面目な話、NIFTY Serveでもどこにでもアクセスして使用できます。

このプログラムはRS-232C、つまりモデムからデータが送られてくると画面にこれを表示します。そして、X68000のキーボードから入力があると、モデムにデータを送ります。

非常に簡単なので、これからプログラミングを勉強したいという方は肩ならしに解析してみるとよいでしょう。プログラムとは難しく考えなければ簡単なものです。

言語はXCのver.2を使用しました。

コンパイルのオプションを次に示します。以後のプログラムも同じオプションです。

A>CC /O /Y ファイル名

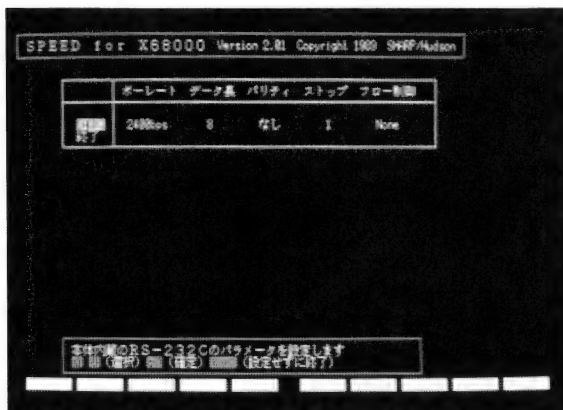
実行するときは次のようにしてください。

A>ETERM

これで、X68000が通信端末になります。終了するとき、ESCキーを押してください。

ちなみに今回は、使用言語にX-BASICを使用する予定でした。具体的には、モデムの制御部分を作り、実際のファイル転送は7月号で作ったプログラムを内部から呼び出すつもりでした。

しかし、RS-232Cのデータ受信をする際、何バイト送られてきたかの確認をX-BASICでは素直にできないとのことなので（シャープのサポートの方に教えていただ



画面1 SPEED.Xの設定

きました)、従来どおりC言語で記述しました。もし、皆さんが通信制御プログラムを作ろうと思いたったならば、C言語の採用をすすめます。XCには非常に豊かな制御機能が用意されておりこちらのほうが結果的に楽です。

## モデム&通信の設定

この通信ソフトは、設定をHuman68kに依存しています。SPEED.Xにて画面1のように設定してください。

おそらく、ここ2年以内に発表された国産のモデムであるならば、この通信ソフトはこの設定だけで動くと思います。

ですが、古いタイプのモデムなどで現在の標準的な機種と設定の異なるモデムの場合動作しないことが考えられます。ここではモデムの設定を説明します。動かないときはお手持ちのモデムの説明書をよく確認してください。

さて、話を整理するために接続の概要を図3に示します。各機材の継目で固有の設定が考えられます。図よりそれぞれの勘どころを押さえてみてください。

ちなみに、筆者は今回、図4の環境&設定で実験をしています。

## 戯れにアクセス

モデム制御に慣れる意味でETERMよりモデムを制御

## MNPとは

現在のモデムはMNPという方式がほぼ標準となっています。MNPとは米マイクロコム社の提唱した規格で現在クラス5、ないし10と呼ばれるものが回回っています（MNPはMicrocom Network Protocolの略）。

この規格は、モデム↔公衆回線↔モデムの間での接続を決めたものです。

電話回線を介してデータを送るとどうしてもデータが化けてしまいます。

MNPはこれを補正します。ですから、送信側と受信側でMNPを上手に使用すればノーエラーを実現できます（当然、双方MNP対応の必要があります）。

さて、MNPも規格が高度化しています。クラスが上がるにつれ、性能が高くなると考えてください。

MNP Class4:ノーエラーを実現している。

MNP Class5:ノーエラーのうえに、データ圧縮機構を採用し最高300%転送速度を上げています（カタログには記載）。

MNP Class10:移動体通信、つまり携帯電話、自動車電話を前提とした、非常に回線状態の変化が激しい環境でもエラーフリーを謳った規格。しかし、通常電話回線より、デジタルの移動体のほうが回線品質がよいので不要では、という指摘もあり。

MNPは、モデム同士が接続するときは相手のクラスを確認し、低いほうに合わせることで異なるクラス同士でも接続することができます。

通常は、クラス4ないし5を利用すればよいでしょう。



図3

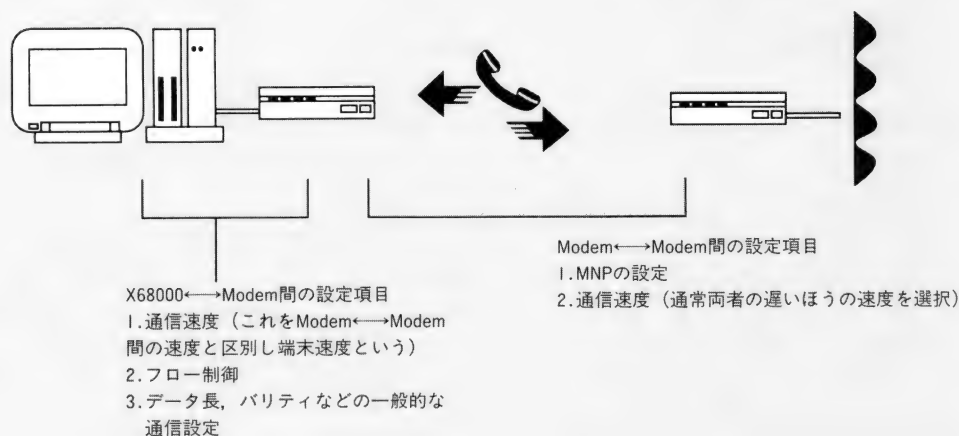
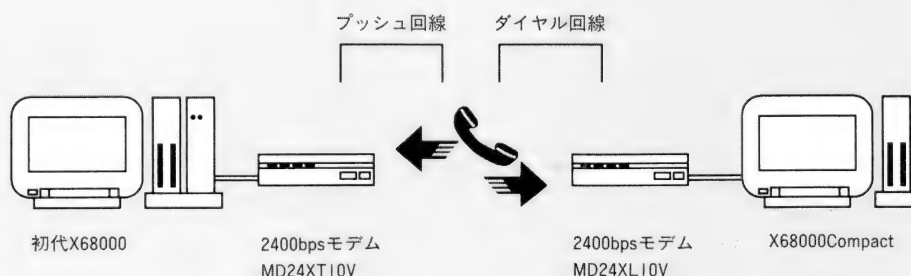


図4



するATコマンドを実行してみましょう。

代表的なAT命令として表1のようなものがあります (というよりあとはあまり使わない)。ここでは、オムロンの取扱説明書を基にしています。メーカーにより★のついた命令は異なる場合があります。注意してください。

これらは、ATZとAT&F&WそしてATDを除き複合実行できます。つまり、AT¥J0とAT¥Q2の27をあわせて“AT¥J0¥Q2”のようにすることができます。

通常は、次のような命令を発行し電話をかけます。

```
ATZ
OK
ATX3¥J0¥Q2¥N3
OK
ATD03-3447-2564
CONNECT
```

ATDコマンドで、相手の電話番号を指定しダイヤリングします。そして、接続に成功するとCONNECT (通話中ならBUSY) というメッセージがモデムより返され、データモードになります。

データモードとは、X68000から送ったデータをすべて相手に送るモードです。このモードになると、ATコマンドは発行できません。発行しても、モデムはATコマンド

とは認識せず、相手に送る通信データとして処理します。

東京のNIFTY Serveにアクセスする様子を画面2に示します。

さて、注意事項として、ETERMはプログラム終了時に

表1

命 令	機 能
ATZ	モデムを初期化する。初期化には3秒時間がかかる
ATXn	相手モデムと連結時のメッセージを指定する。 nは数字。通常は1ないし3を指定
AT¥J0	端末速度固定モード。Modem $\longleftrightarrow$ Modem間の速度とは別にX68000 $\longleftrightarrow$ Modem間の速度を指定できる。
AT&F&W	モデムを工場出荷状態に初期化する
★AT¥Qn	フロー制御を指定。nは番号。0=なし 1=Xon/Xoffフロー, 2=ハードフロー
★AT¥Nn	MNPを指定。nは番号。0=なし, 3=MNPクラス5指定
ATD電話番号	続く番号のところへ電話をかける
ATS0=n	自動着信するまでのベルの回数を指定。n=0で自動着信禁止。nは回数
AT%R	内部レジスタ (設定メモリ) を一覧表示
+++	モデムをコマンドモードにする
ATH0	回線切断

回線切断ということを特にしていません。ですから、終了時に意識的にモデムの電源を切ることをおすすめします。

## 本題

閑話休題，基礎実験のところですから手間どりました。ここからが本題です。

ここまでで電話回線を使った通信というものの概略が体感できてきていると思います。要はファイル転送する前処理として，モデムの初期化とダイヤリング処理を挿入すればよいということなのです。

これはまた，受信側にあてはめるならば，同様にモデ

ム初期化を行い，電話がかかってきたならば電話を受けるという処理を入れれば，あとはファイル受信の処理をすればよいということです。

### ●受信

ここで，ひとつ問題が出てきます。我々は通常，パソコン通信などで電話をかけるということは行っています。また，前の章でもダイヤリングの実験を延々してきました。

さて，受信する側，いい換えるならば電話を受ける側の動作はまだ実験していません。これをクリアしなければいけません。結論からいいますと，可能です。モデムを初期設定するときに自動受信モードを指定すれば，電話がかかってきた時点で自動的に受けてくれます。この設定はATS0=n コマンドにて設定します。nには，着信時に何回ベルが鳴ったら受けるか回数を指定します。ここにゼロを指定すると自動着信禁止になります。実際には，次のようにすればよいのです。

ATS0=1

これはまた複合命令として設定できますので，プログラム中より，次のように組み合わせて使うのが一般的でしょう。オムロンのMD24XL10VでMNPクラス5を利用しハードウェアフローにて交信するものとしましょう（オムロン製であれば，よほど古い機種でなければそのまま使えます）。

ATZ

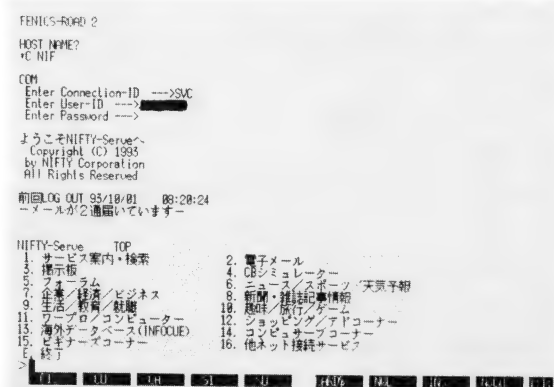
OK

ATS0=1X3¥J0¥Q2¥N3

### ●簡単な実験

ETERMを2台のX68000の上で実行。そして，一方から残りへ電話をかけてみました。そして，双方でESCキーを押しETERMを終了させます。

ここで，ETERMは終了時に回線切断などを特に行わないことを思い出してください。つまり，2台のX68000は，いまだにモデムを介しRS-232Cで接続されたままの状態です。そして，このときモデムは，データモードですから，クロスケーブルで直結したのと同じ状態といえます。



画面2 ETERMで通信をしているところ

## リスト2 TR.C

```
1: #include <stdio.h>
2: #include <stdlib.h>
3: #include <iocslib.h>
4: #include <time.h>
5:
6: void _time_set();
7: void _time_st();
8: void _rs_buf_clr();
9: void _rs_echo();
10:
11:
12:
13: #include "tl.h"
14:
15:
16: void main( argc, argv ,
17: int argc;
18: char *argv[];
19: {
20:     int c;
21:
22:     _rs_buf_clr();
23:
24:     if( modem_set( argc, argv ) ) {
25:         printf( "モデム初期化に失敗しました\n" );
26:     }
27:     else {
28:         _rs_buf_clr();
29:         system( "TENRSI" );
30:         _time_set(1);
31:         _rs_puts( "+++ATH0" ); /* 回線切断 */
32:     }
33: }
34:
35:
36: /**/
37: /*****
38: * modem_set      モデムを初期化しダイヤリングする
39: * sts = 0 正常出力、モデムより"OK"が戻る
40: *          1 "CONNECT..."
41: *          2 "BUSY"通話中
42: *          -1 その他の場合。エラーとみなす
43: *****/
44: int modem_set( argc, argv )
45: int argc;
46: char *argv[];
47: {
48:     int sts;
49:     char wk[62];
50:
51:     sts = -1;
52:
53:     _time_set( 4 ); /* ATZコマンドの初期化時間は3秒、余裕をもって4 */
54:
55:     if( _rs_puts( "ATZ" ) ) {
56:         printf( "モデムの初期化に失敗しました\n" );
57:     }
58:     else if( _rs_puts( "ATS0=1" ) ) {
59:         printf( "モデムの初期化に失敗しました\n" );
60:     }
61:     else {
62:         sts = 0;
63:         if( argc>2 ) { /* モデム初期化指定文字列あり */
64:             if( sts=_rs_puts( argv[1] ) ) {
65:                 printf( "モデムの初期化に失敗しました\n" );
66:             }
67:         }
68:     }
69:
70:     return( sts );
71: }
```



## ダイヤル電話とプッシュ電話

モデムの設定でまず間違えるのがこれです。

電話交換機には新旧で2種類の方式があります。

古いものでダイヤル式(パルス式ともいう)。新しいものでプッシュ式(トーン式ともいう)。この両者の違いは、電話をかけるときの相手の電話番号の指定方法の違いにあります。

交換機は発信者の電話を、ダイヤリングした番号(つまり接続先の電話番号)につなぎます。

この番号は、古いダイヤル式では電気信号のパルスの数で知らせます。モールス信号のようなかたちで送ります(図5)。

昔懐かしい黒電話のぐるぐる回る円盤は実は、あのパルスを機械的に発生させるためのものだったのです。円盤が回転するときに数字の穴の数だけ、パルスを発生させるようになっていたわけです。

プッシュ式は押しボタンに特定の周波数(音と違ってください)を割り振り、電話番号を交換機に送る方式です。

まぎらわしいのですが、現在の電話機はほとんどが押しボタン式でみかけはプッシュです。ですがプッシュ/パルス両用となっており、どちらの機能も備えています。これはモデムも一緒です。ディップスイッチなどで設定するようになっていますので、利用するときはよく確認しましょう。

図5



## リスト3 T1.H

```
1:
2: /**/
3: /*****
4: * _rs_puts 文字列に改行をつけてRS232C出力
5: *          sts = 0 正常出力、モデムより"OK"が戻る
6: *          1 "CONNECT..."
7: *          2 "BUSY" 通話中
8: *          -1 その他の場合。エラーとみなす
9: *****/
10: int _rs_puts( s )
11: char *s;
12: {
13:     int i;
14:     int sts;
15:
16:     i = strlen( s );
17:     i++;
18:
19:     sts = -1;
20:
21:     while( *s ) {
22:         OUT232C( (int)*s );          /* 1文字出力 */
23:         s++;
24:     }
25:
26:     OUT232C( 0x0d );                /* 改行出力 */
27:
28:     _time_st();
29:     while( _time_over() == 0 ) {
30:         if( LOF232C() >= i ) {
31:             _rs_echo( i );
32:             sts = _rs_rts();
33:             break;
34:         }
35:     }
36:
37:     return( sts );
38: }
39:
40: /**/
41: clock_t _time_len;
42: /*****
43: * _time_set タイムアウト時間をセット
44: *****/
```

Human68kの提供するAUXがそのまま使えます。ですから、この状態で相手方にCOPYコマンド等を利用してファイルを転送できます。7月号で紹介した、lha、ishを利用したBATプログラムをそのまま使えるはずですよ。

理由はわかりませんが、SPEEDコマンドでの設定を9600bps設定したときはうまくいかず、2400bpsで動作確認したことを添えておきます。

### ●資源の再利用

さて、ここで熟考。モデムを使ったファイル転送のやり方がわかりました。しかしここで新しく、ファイル転送プログラムを作るのではなく、すでにあるプログラムをそのまま利用できないでしょうか。

C言語の中にsystem関数という機能が用意されています。これはなにかというとコプロセッサと呼ばれるものです。これを使うと、Cのプログラムの中から、COMMAND.Xを呼び出すことができます。もし、COMMAND.Xを呼び出せばその中から、通常アプリケーションを実行できます。

7月号で作ったファイル転送プログラムをここで実行できれば、あとはモデム処理だけを考えればよいはずですよ。

またCOMMAND.Xを利用するわけですから、DIR命令なども使えて好都合です。

### ●受信プログラム

リスト2が受信プログラムTRです。リスト3にインクルードを示します。

処理としては、モデムを初期化します。そして、電話がかかってくるのを待ちます。自動受信では、電話が鳴っているときにはモデムはX68000に対して、“RING”という文字列を送って知らせます。そして、自動受信(受話器を取る動作)した時点で、“CONNECT”という文字を送ります。ですから、この文字を監視すればいいわけです。厳密には“CONNECT 2400”のように、接続した通信速度などをあわせて知らせてきますから、先頭の“CON”および、文字列の最後についてくる改行コードを監視すればよいでしょう。

もっとも、7月号のTENRSIは、TENRSOから送信開始コードがくるまでは、ほかの受信データをゴミとみなし読み捨てています。ですから、極端な話、TENRSIを直に実行しても、受信ができてしまいます。これは、テストをしていて気がついたのですが、筆者自身驚きました。

というわけで、TRは実際には、回線初期化、TENRSIの実行、そして回線切断の3プロセスよりなります。

実行するにあたり必ず同じディレクトリに7月号で作ったTENRSIを入れておいてください。

使用方法是次のとおり。

A>TR モデム初期化命令

モデム初期化命令はオプションで省略できます。通常は、

```

44: *****/
45: void _time_set( s )
46: int s; /* 秒数 */
47: {
48:     _time_len = (clock_t)(s * CLK_TCK); /* 秒をシステム時間に変換格納 */
49: }
50:
51:
52: /**/
53: clock_t _time_out;
54: *****/
55: # _time_st タイムアウトカウント開始
56: *****/
57: void _time_st()
58: {
59:     _time_out = _time_len + clock();
60: }
61:
62:
63: /**/
64: *****/
65: # _time_over タイムアウトチェック
66: # sts = 0 時間内
67: # 有値 タイムアウト
68: *****/
69: int _time_over()
70: {
71:     int sts;
72:
73:     sts = 0;
74:     if( _time_out < clock() ) {
75:         sts = -1;
76:     }
77:
78:     return( sts );
79: }
80:
81:
82: /**/
83: *****/
84: # _rs_buf_clr 受信バッファを空にする
85: *****/
86: void _rs_buf_clr()
87: {
88:     int c;
89:
90:     while( LOF232C() ) { /* 受信文字はあるか? */
91:         c = INP232C(); /* Yes.& 読み取り */
92:     }
93: }
94:
95:
96: /**/
97: *****/
98: # _rs_echo 受信バッファを読み表示
99: *****/
100: void _rs_echo( n )
101: int n; /* 読み取る文字数 */
102: {
103:     int c;
104:     int i;
105:
106:     for( i=0 ; i<n ; i++ ) {
107:         c = INP232C(); /* Yes.& 読み取り */
108:         printf( "%c", c );
109:     }
110: }
111:
112:
113: /**/
114: *****/
115: # _rs_rts モデム応答コードのチェック
116: *****/
117: int _rs_rts()
118: {
119:     int sts;
120:     int c;
121:     int mode;
122:
123:     mode = 0;
124:     sts = -1;
125:
126:     while( _time_over()==0 ) {
127:         if( LOF232C() ) {
128:             c = INP232C();
129:
130:             if( mode==0 ) {
131:                 printf( "%c", c ); /* 表示 */
132:
133:                 if( c=='O' ) {
134:                     sts=0;
135:                     mode = -1;
136:                 }
137:                 else if( c=='C' ) {
138:                     sts=1;
139:                     mode = -1;
140:                 }
141:                 else if( c=='B' ) {
142:                     sts=2;
143:                     mode = -1;
144:                 }
145:                 if( c==0x0a ) {
146:                     ;
147:                 }
148:                 else if( c==0x0d ) {
149:                     ;
150:                 }
151:             }
152:             else {
153:                 printf( "%c", c ); /* 表示 */
154:                 if( c==0x0a ) {

```

A>TR<sup>④</sup>

のように使用すればよいでしょう。モデム初期化命令をオムロンMD24XL10を例に指定するならば次のようになります。

A>TR ATX3YN3YQ2<sup>⑤</sup>

## ●送信プログラム

リスト4が送信プログラムTSです。

使用方法は、まず、相手先で受信プログラムTRを動かします。そして、TSを次のように実行してください。

A>TS 電話番号 モデム初期化命令<sup>⑥</sup>

モデム初期化命令は省略できます。具体的には次のようになります。

A>TS 03-3447-2564<sup>⑦</sup>

そして、向こうにうまく接続できたならば、TSはCOMMAND.Xそのものを内部で呼び出します。

このときには相手のX68000とはRS-232Cがすでにモデムを介して結合しています。

ですから、ここで、7月号で作ったTENRSOプログラムを実行すれば、送ってしまう(?)わけですが。

具体的には、次のようにしてください。試しに、ルートにあるAUTOEXEC.BATを送ってみます。

A>TENRSO A:¥AUTOEXEC.BAT<sup>⑧</sup>

A>TENRSO -E<sup>⑨</sup>

A>EXIT<sup>⑩</sup>

2行目の“A>TENRSO -E”は受信側で動いているTENRSI受信プログラムを終了させるために実行します。EXITは、TSが起動したCOMMAND.Xそのものを終了させるためのものです。これを実行して初めてTSは終了します。

## 次回予告

次回よりいよいよ、拡張FDコネクタを使った実験と研究を行う予定です。

まず、拡張FDがはたして使用できるかの基礎テストを行います。現在、X68000同士を拡張FDコネクタで接続すると、それぞれ相手のFDを増設ドライブとして利用できることが判明しています。

もしも、プログラムから拡張FDコネクタを制御するのであれば、このあたりが障害になってくると考えられます。Human68kはすでに、接続先のFDDを認識、そしてディスクとして使用しているわけですから、これをなんらかのかたちで解除する必要があるでしょう。

そして、これとは別に、デバイスドライバの研究/開発をしていきます。RS-232Cであれ拡張FDコネクタであれ、接続先の記憶装置を仮想ディスクとして使うためにはこの技術が不可欠です。

というわけで、デバイスドライバと拡張FDコネクタの基礎研究を軸に展開していきます。



今回記事を書いていてコンピュータ技術を説明する難しさを感じました。日常なにげなく使っているモデムひとつとってもさまざまな規格、設定が錯綜しています。これはとにもかくにも、コンピュータが巨大で未成熟な複合技術の産物ということを表しているのであると思います。

以後の記事のなかでも、できるだけ低層の基礎技術、知識を紹介していきたいと思います。

#### ●参考文献

別冊トランジスタ技術SPECIAL No.8「データ通信技術のすべて」、CQ出版

C CompilerPRO-68K「C ライブラリマニュアル」、シャープ

## パソコン通信端末同士の際の問題

通信ソフトを利用しパソコン同士をつなぐのは非常にやっかいです。

理由はいろいろありますが、結局、ソフトがそういう使い方を想定していないということだと思います。

パソコン通信ソフトはあくまで自動運転されているホストにアクセス、対話式に操作するように作られているということでしょうか。

これは、実際にパソコン同士を電話でつないでみればわかります。アクセスするまでは簡単ですが、そのあとが大変です。

これを示すと図6のようなことをしますが、なにしろ相手は遠隔地です。見えない相手が対象ですので非常に面倒です。ただし、例外的にファイルの自動受信機能を備えた通信ソフトであれば快適にいくはずです。

現在、X、Y、ZMODEM、Transit、B-Plus、Quick-VANという主要転送方式があります。筆者の知る限り、自動受信を信頼性よく実現できるのはZMODEMとTransitだけです。

もし、受信側にどちらかの自動受信機能があれば現実的な方法になるでしょう。

まあ、マクロ言語搭載の通信ソフトであれば自動受信マクロをプログラミングすればできるかもしれません。まず不可能と思ったほうがよいと思います。

#### XMODEM

この方法はファイル名を受信側に送れないから不向き。それに、受信側が転送開始の合図を発行するので自動受信は考えられない

#### YMODEM

これはXMODEM同様、受信側が転送開始の合図を発するので自動受信は考えられない

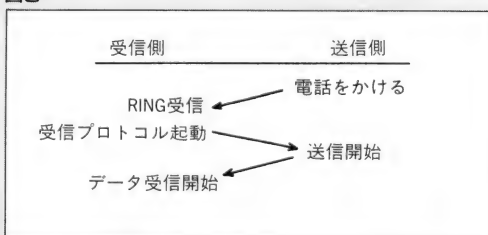
#### B-Plus

根本的に端末同士で接続できない仕様になっている

#### Quick-VAN

X、YMODEM同様、受信側が転送開始の合図を発するので自動受信は考えられない

図6



```
155: break; /* IF code終了 */
156: }
157: }
158: }
159: }
160: return( sts );
161: }
```

## リスト4

## TS.C

```
1: #include <stdio.h>
2: #include <stdlib.h>
3: #include <ioclib.h>
4: #include <time.h>
5:
6: void _time_set();
7: void _time_st();
8: void _rs_buf_clr();
9: void _rs_echo();
10:
11:
12: #include "tl.h"
13:
14:
15: void main( argc, argv )
16: int argc;
17: char *argv[];
18: {
19:     int c;
20:
21:     _rs_buf_clr();
22:
23:     if( argc<=1 ) {
24:         printf( " TS 電話番号 <モデム初期化命令>YnYn" );
25:     }
26:     else if( modem_set( argc, argv ) ) {
27:         printf( " モデム初期化に失敗しましたYn" );
28:     }
29:     else {
30:         system( "" );
31:         _time_set();
32:         _rs_puts( "+++ATH0" ); /* 回線切斷 */
33:     }
34: }
35:
36:
37: /**/
38: /***** モデムを初期化しダイヤリングする *****/
39: # modem_set
40: # sts = 0 正常出力、モデムより"OK"が戻る
41: # 1 "CONNECT..."
42: # 2 "BUSY" 通話中
43: # -1 その他の場合。エラーとみなす
44: /*****
45: int modem_set( argc, argv )
46: int argc;
47: char *argv[];
48: {
49:     int sts;
50:     char wk[62];
51:
52:     sts = -1;
53:
54:     _time_set( 4 ); /* ATZコマンドの初期化時間は3秒、余裕をもって4 */
55:
56:     if( _rs_puts( "ATZ" ) ) {
57:         printf( " モデムの初期化に失敗しましたYn" );
58:     }
59:     else {
60:         if( argc>=3 ) { /* モデム初期化指定文字列あり */
61:             if( _rs_puts( argv[2] ) ) {
62:                 printf( " モデムの初期化に失敗しましたYn" );
63:             }
64:         }
65:
66:         _time_set( 40 ); /* 電話接続は応答まで30秒以上かかる */
67:
68:         strcpy( wk, "ATD" );
69:         strcat( wk, argv[1] );
70:         sts = _rs_puts( wk );
71:         switch( sts ) {
72:             case 0:
73:                 printf( "なんかモデムかへんてすYn" );
74:                 sts = -1;
75:                 break;
76:             case 1:
77:                 printf( "回線接続完了送信を開始してくださいYn" );
78:                 sts = 0;
79:                 break;
80:             case 2:
81:                 printf( "通話中てすYn" );
82:                 break;
83:             default:
84:                 printf( "エラーてんねんYn" );
85:         }
86:     }
87:
88:     return( sts );
89: }
```



(で)のショートプロばーてい——(その50)

# オモイコンダラ・プ2グラム

Komura Satoshi 古村 聡

今月のショートプロはゲームにBASICの関数、ツールと盛りだくさん。特にツールは一見の価値あり。X-BASICだけで動くのでみんな楽しめるぞ。ちょっと疲れ気味の(で)さんに皆さん励ましのお手紙よろしく！ 質問も待ってまーす。



illustration : T.Takahashi

私は思い込みがはげしいぞー、文句あるかつ！

常日頃から「思い込みはげしすぎ」「変なヤツ」などといわれる私なであります(はげしいはげしくないはともかく、変なヤツは余計だと思ふぞ)。思い込んだら命懸け、とことん思い入れちゃうってーのはやっぱりゲーム作り&ゲームをする人としては実はとっても正しい姿勢なんではないかと思うのですよね。

パソコンゲームっていうのは、どんなにリアルにしようとかんばってもディスプレイ上に描かれた絵でしかない。そこにのめり込むには想像力と思い込みしかない。

まして、プログラミングをするってえことは、その先に書いては直し、書いては直しのデバッグ作業が待っているわけで、いくら書いても直らない、押しても引いても動かない、3日たっても完治しない、そんな地獄の何日間かをすごすんだとしたら、好きでなければやってられないですよね、ほんと。

そんなわけで、好きなんだから好きなんだから好きなんだからいいじゃないかいじゃないかいじゃないかい！ という姿勢が大事なんではないかと思うんです私は、はい。

そーゆーわけで、アンミラのスカートは短くなくっちゃだめだ！ 亜美ちゃんは世界一かあいぞ！ 「ああっ女神さまっ」のLDは3巻だけは絶対買いだぞ。女神3姉妹の末の妹神、スクルド様は凶悪にかあいくていいぞ！ ハアハアハア。



## 愛と正義のカーレース

さーて、ではでは今月のトップバッターですね。まずはHCグランプリの作者の中村さんのプログラムでHCグランプリのパート2、HC2.Cです。どうぞ～。

HC2.C for X680x0

(要Cコンパイラ、ジョイスティック)

東京都 中村俊之

こらこら、そこそこっ！ 「なーんだ、ただのパート2か」などとみくびってはいけません。今回はCで書かれてグレードアップして帰ってきたんです。

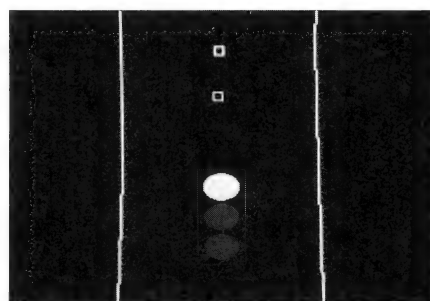
このプログラムはCで書かれていますから、遊ぶにはエディタでリストを入力して、CコンパイラPRO-68K ver.2.0以上か、GCCを使って実行ファイルを作ってくださいね。その際、ちょっと重いプログラムなので利用できる環境であれば、GCCのほうでコンパイルしたほうが良いと思います(GCCの場合はXVI以上のマシンを使っただけでいいですね)。コンパイルするときには、BASICやIOCSのライブラリを使いましょうから忘れないでください。

それでもって遊び方。このHC2は前作と同じように、コースエディタで作って、自分のコースで遊ぶカーレースゲームです。無事HC.Xができれば、

A>HC2 コースファイル名 プレイヤー名  
でプログラムを実行しましょう。このときコースファイルがあった場合には、そのコースでゲームが始まり、ない場合にはコースエディタが起動します。

コースエディタはマウスで操作します。まず、スタート位置からゴールまで外周を引いて、それから内周を引きます。そして、ゴールをHCカーのスタート地点に置き、最後にフラグをゴールと反対側のコース上に置きます。コースエディタには現在の書き込みモードが表示されますから、それをよく見ながら作っていきましょう。

[外周] ならば外周を描くモード、[内周] ならば内周を描くモードになっています。それから、[goal]では外周と内周の始点より少し前に、コースよりやや大きめにBOXを描き、[flag]ではコース中間点、つまり



HC2.C

ゴールの反対側にBOXを描きます。

コースができたら、HCカーで走ることができます。では、ばーつと遊びましょう！

基本的なルールは前作HCと一緒に、ジョイスティックを使って過去の自分の走り(ライバル車)に追いつき追い越せ、ゴーゴーっ！ ってわけ。ちなみに操作はスティックで左右、ボタンA、Bがアクセルとブレーキになっています。

いや～、ずいぶんグレードアップしましたね～！ 前作のHC.BASに比べると画面の表示も拡大されてるし、ほとんど別のゲームになってます。前回「プログラムにアラが目立つ」って書かれて発奮して作ったんですけど、そーか、そんなにあのプログラムが好きだったのだね。こんなに立派になるとは私、全然予想できませんでした、はい。やはり、HCグランプリに対する愛のなせる技ですね、うんうん(いっててちょっとはずいぞ)。

ただ、個人的には私、CよりはBASICのゲームのほうが好きなんですけどね。打ち込み楽だし。

前作のHC.BASと一番違うのは、作ったコースがゲーム中は8倍に拡大して表示されることなわけです。つまりコースを作るときに、うまく1/8のサイズで描いてそのコースの幅がゲーム中にぴったりくるかどうかゲームの面白さの分かれ目なんです。前作もまともに遊ぼうとするとコース作りが結構むずかったけど、今回はさらに輪を



かけて努力が必要です。ま、愛さえあれば  
へーきなんでしょーけど(今回は全部これ  
で片付けてしまおう)。

ところでですね。このプログラムなんて  
すけど、ちょっとリストを詰めすぎですよ  
ー。確かにショートプロっていうのは短い  
ほうがいいんですけど……うーん、やっぱ  
りリストは見やすいほうが良いというのも  
事実なんでありま。ましてこのゲームの  
場合は120行ちょっとなんだし、読みにくく  
なるまで詰めないでくださいねー。お願い  
(いっておいたら、また改良版が送られてく  
るんだらうか……。期待して待てよう)。



## スプライトに便利!

さてさて、続きましては今月2本目のプ  
ログラム。グラフィック画面に描いたもの  
をそのままスプライトにしてくれちゃう  
BASIC/Cコンパイラ用外部関数EXSPR.  
FNCなのです。どうぞ!!

### リスト1 HC2.BAS

```
1: #include<ioclib.h> /*
2: #include<doslib.h> /* H C C A R G R A N D P R I X 2
3: #include<stdio.h> /*
4: #include<math.h> /*
5: #include<basic0.h> /*
6: #include<graph.h> /* 1993 おく の 模 擬 運 動 会
7: #include<mouse.h> /* OBU allrights reserved
8: #define U unsigned char
9: #define F float
10: int dat_ld(void);int dat_wt(void);int ed(void);void m(void);
11: void cswt(void); void lin_wt(int); void lin_er(int);
12: U *fnp,*ynp,*hnp,*c;
13: U *mes[]={ (U*)"外周", (U*)"外周", (U*)"内周",
14: (U*)"内周", (U*)"goal", (U*)"goal",
15: (U*)"flag", (U*)"flag", (U*)"H C",
16: (U*)"角度" };
17: short *gad=(short *)0xd80000, sr[]={ 0,0,1984,1984,63 };
18: int cs_dt[1000],cxy[2][500],*pc=cs_dt,c,mod,x,y,btn,fno,
19: size,page, st, me_dt[10000],*mp,tim[4],hi_dt[10000],
20: *hp,hi,hx,hy,exy[2][1000],*ep,*sp[2],r,rm,joy,
21: rap,rf[3],t;
22: float rs,rc,rw,ms,my,s,gb[4],fb[4];
23: void main(int gc,char *gv[]) {
24: st=0; if( gc == 3 ) fnp=(U*)gv[1], ynp=(U*)gv[2];
25: else( printf("usage:hc2 filename yourname\n"); goto XT; )
26: if( dat_ld()==-1 ) { printf("disk error.\n"); goto XT; }
27: while( strig(1) != 1 ) { /*----- main loop */
28: cswt(); CRTMOD(0x100+6); mp=me_dt; hp=hi_dt; r=rm; s=0;
29: for( c=0; c<3; ++c ) rf[c]=0,tim[c]=0,rap=0;
30: rs= F sin(F r*F 0.0174); rc= F cos(F r*F 0.0174);
31: exy[1][2]= 0xffff; ap[0]= &cxy[0][0]; page=0; btn=1;
32: exy[1][5]= 0xffff; ap[1]= &cxy[1][0]; m[1]; btn=1;
33: for( c=2; c<5; ++c ) for( x=0; ++x<16000; y=printf(""); )
34: GPALET(c-1,14798 ); GPALET(c, sr[c]);
35: while(rap<3) m(); GPALET(5,31*2); VPAGE(8); CRTMOD(260);
36: C_LOCATE(17,16);PRINT("H C C A R G R A N D P R I X");
37: C_LOCATE(24,20);printf("Top %s:%d",hnp,hi);
38: C_LOCATE(24,22);printf("Your Time:%d",t);
39: while( strig(1)!=0 ) printf(""); while( strig(1)==0 )
40: if( hi>t ) { for(c=0,hi=t; c<hi*2; hi_dt[c]=me_dt[c++]);
41: *hnp=*ynp; (*hnp+1)=*(ynp+1); (*hnp+2)=*(ynp+2); }
42: CRTMOD(16);C_CURON();C_FNMKMOD(0);dat_wt(); XT=0;
43: void m(void) { /*----- race */
44: if( (joy==stick(1))&& 4 ) rs= F sin(F r+3)*F 0.0174;
45: rc= F cos(F r+3)*F 0.0174;
46: else if( joy == 6 ) rs= F sin(F r-3)*F 0.0174;
47: rc= F cos(F r-3)*F 0.0174;
48: if( (joy==stick(1))&& 2 ) { if( btn==0 ) s += 0.17; }
49: else if( joy==1 ) { if( s == 0.5 ) s=0; }
50: else if( btn==0; if( s == 0.1 ) s=0; }
51: mx=-s*rs; my=s*rs; (*mp++)=(int)mx; (*mp++)=(int)my;
52: if( t == 35 ) for( c=1; c<5; ++c ) GPALET(c, 0 );
53: if( ++t<hi ) { APAGE(2); box(hx-2,hy-2,hx+3,hy+5,0,65535);
54: hx=128+(int)(F((hp-(int)mx)*rc)-F((hp+1)-(int)my)*rs));
55: hy=220+(int)(F((hp-(int)mx)*rs)+F((hp+1)-(int)my)*rc));
56: box( hx-2,hy-2, hx+3,hy+5, r, 65535 ); hp=2;
57: box( 125,218, 130,225, 15, 65535 ); }
58: if( (gad+(int)(my/4)*512+(int)(mx/4))&& s>3 ) s=3;
59: if( mx>gb[0] && mx<gb[2] && my>gb[1] && my<gb[3] &&
60: rf[rap]==1 ) printf("Lap%ld",rap,tim[rap++]);
61: else if( mx>fb[0] && mx<fb[2] && my>fb[1] && my<fb[3] )
62: rf[rap]=1; C_LOCATE( 0, rap ); }
63: if( ++page == 2 ) page=0;
64: APAGE(page); lin_er(&exy[page][2]); ep=&exy[page][0];
```

## EXSPR.FNC for X-BASIC

(要アセンブラ, リンカ)

神奈川 松本岳美

この外部関数パッケージには、グラフィ  
ック画面に表示した絵をスプライトデータ  
として定義する関数grsp\_def()と、1つの  
命令でたくさんのスプライトを表示できる  
SP\_BROCK()命令を含んでいます。

この関数を使うためにはアセンブラとリ  
ンカが必要です。Cコンパイラのパッケー  
ジに含まれるAS.X/LK.Xがあるいは  
X68k Develop.に含まれるHAS.X/HLK.  
Xなどを用意してください。

このプログラムには4つのリストが掲載  
されています。それぞれBASIC用の外部関  
数の中身(リスト2)、コンパイラ用のgrsp  
\_defの関数定義(リスト3)、sp\_brockの関  
数定義(リスト4)、サンプルプログラム(リ  
スト5)になります。コンパイラにも使う  
場合は全部必要ですが、BASIC中で使うだ  
けならリスト2だけでOKです。

さて、それではリスト2を、エディタで  
入力してください。エディタは付属のED.X  
や、通なところではフリーウェアのμEmacs  
など、なにを使ってもかまいません。

それからコマンドライン上から、

A>AS EXSPR.S

A>LK EXSPR.O

もしくは、

A>HAS EXSPR.S

A>HLK EXSPR.O

としてアセンブル、リンク作業をして、

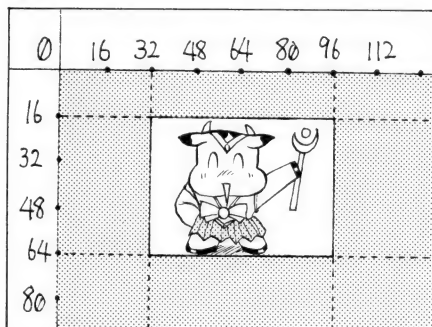
A>REN EXSPR.X EXSPR.FNC

としてファイル名を変えればこれで完成で  
す。

私が9月号で「スプライト定義を2行で  
すます関数があれば完璧なんだけどねっ」  
と書いたら、この作者の松本さん、「それ  
なら」ってんで送ってくれました。いって  
みるもんですね。グラフィックさえ1行  
で描ければ本当に2行で定義できちゃいま  
すねー。

```
65: lin_wt( 0 ); lin_wt( 1 ); VPAGE( page+13 ); ++tim[rap]; }
66: void lin_wt( int mod ) { /*----- line-writer */
67: int *p=sp[mod], ox,oy,x,y, xx=(p++)-mx,yy=(p++)-my;
68: x=128+(int)(F(xx*rc)-F(yy*rs)); *(ep++)=x;
69: y=220+(int)(F(xx*rs)+F(yy*rc)); *(ep++)=y;
70: for( c=0; c<7; ++c ) {
71: if( *p == 0xffff ) p=&cxy[mod][0];
72: xx=(p++)-mx; yy=(p++)-my; ox=x; oy=y;
73: x=128+(int)(F(xx*rc)-F(yy*rs)); *(ep++)=x;
74: y=220+(int)(F(xx*rs)+F(yy*rc)); *(ep++)=y;
75: line( ox,oy,x,y,15,65535 );
76: if( c==1 && y>270 ) if( *(sp[mod]+2)==0xffff )
77: sp[mod]=&cxy[mod][0]; *(ep++)=0xffff; }
78: void lin_er( int p ) { /*----- line-eracer */
79: while( *p != 0xffff ) { p++;
80: line( *(p-4),*(p-3), *(p-2),*(p-1),0,65535 ); }
81: p++; while( *p != 0xffff ) { p++;
82: line( *(p-4),*(p-3), *(p-2),*(p-1),0,65535 ); } }
83: int dat_ld( void ) { /*----- data-loader */
84: CRTMOD(4); G_CLR_ON(); hi=10000; mod=0; c=0;
85: if( (fno=OPEN( fnp,0x002 )) > 0 ) {
86: READ(fno,&size,4); READ(fno,cs_dt,size*4);
87: READ(fno,&hi,4); c=READ(fno,hi_dt,hi*2*4);
88: if( c == hi*2*4 ) CLOSE( fno ); else return(-1);
89: } else size=ed(); return( size ); }
90: int dat_wt( void ) { /*----- data-writer */
91: if( (fno=CREATE( fnp,0x20 )) < 0 ) return(-1);
92: WRITE( fno,&size,4 ); WRITE( fno,cs_dt,size*4 );
93: WRITE( fno,&hi,4 ); WRITE( fno,hi_dt,hi*2*4 );
94: CLOSE( fno ); return( 0 ); }
95: int ed( void ) { /*----- couse-editor */
96: mouse(4);mouse(1);MS_LIMIT(0,511,511);MS_CURST(255,255);
97: while( mod < 9 ) {
98: mpos( &x,&y ); btn=MS_GETDT(); *(p+c)=0xffff; APAGE(0);
99: if( (btn&0xff00) == 0xff00 ) { *(p+c++)=x; *(p+c++)=y;
100: switch( mod ) {
101: case 0: case 2: case 4: case 6: case 8:
102: pset( x,y,15 ); ++mod; break;
103: case 1: case 3:
104: line( *(p+c-4),*(p+c-3),x,y,15,65535 ); break;
105: case 5: case 7:
106: box( *(p+c-4),*(p+c-3),x,y,15,65535 ); ++mod; }
107: } else if( (mod==1 || mod==3) && (x!=hx || y!=hy) ) {
108: APAGE(1); line( *(p+c-2),*(p+c-1),hx,hy,0,65535 );
109: line( *(p+c-2),*(p+c-1),x,y,14,65535 ); hx=x;hy=y; }
110: if( (btn&0xff)==0xff && (mod==1 || mod==3) ) { ++c; ++mod;
111: APAGE(1); line( *(p+c-3),*(p+c-2),hx,hy,0,65535 ); }
112: while( (btn&0xffff) != 0x0000 ) btn=MS_GETDT();
113: C_LOCATE(0,0);printf("Size:%d",size);fnp,c,mes[mod]);
114: mouse(2);scanf("%d",&rm);*(p+c++)=-rm; return( (int)+c ); }
115: void cswt( void ) { /*----- couse-write */
116: C_CUROFF(); CRTMOD(3); CRTMOD(4); G_CLR_ON(); APAGE(3);
117: for(mod=0,c=0,cc=2;mod<2;cc+=c,cxy[mod][cc]=0xffff,cc=2) {
118: cxy[mod][0]=*(p+c++)*4; cxy[mod][1]=*(p+c++)*4;
119: while( *(p+c) != 0xffff ) { GPALET( 5, 0 );
120: cxy[mod][cc++]=*(p+c++)*4; cxy[mod][cc++]=*(p+c++)*4;
121: line( *(p+c-4),*(p+c-3),*(p+c-2),*(p+c-1),5,65535 ); }
122: for( x=0; x<4; ++x ) gb[x]= *(p+c++)*4;
123: for( x=0; x<4; ++x ) fb[x]= *(p+c++)*4;
124: paint(*(p+c),*(p+c+1),5); mx=*(p+c+1)*4; my=*(p+c+1)*4;
125: rm=*(p+c+1); hnp=(U*)(p+c); *(p+c+1)=0; APAGE(2);
126: for( c=2,x=127,y=0,r=26; c<5; ++c ) { GPALET( c,14798 );
127: circle( x,r*c-y,r/2-2,c,0,360,270 ); paint( x,r*c-y,c );
128: GPALET( 1,14798 ); box( x-14,52-15,x+14,104+15,1,65535 ); }
```

図1



このプログラムの使い方なんですが、  
grsp\_def()のほうが、

```
grsp_def(xx,yy,kx,ky,sp)
```

xx .....グラフィック画面上のx座標  
(0~511)

yy .....グラフィック画面上のy座標  
(0~511)

kx .....x方向を定義するパターン個  
数

ky .....y方向を定義するパターン個  
数

sp .....定義を開始するスプライトパ  
ターンコード(0~255)

それから、sp\_brock()のほうが、

```
sp_brock(sp,xx,yy,ch,of,nn,pr)
```

sp .....表示を始めるスプライトプレ  
ーン番号(表示個数分使用)

xx .....xベース座標

yy .....yベース座標

ch .....表示情報を格納したchar型配  
列

of .....その配列変数のいくつ目から  
表示するか

nn .....表示個数

pr .....プライオリティ。BASICのSP\_  
SET()のプライオリティと同じ  
表示情報は、

spr(0) : 1個目のスプライトのブロッ  
クの中の相対x座標

spr(1) : ブロックの中の相対y座標

spr(2) : 反転情報&パレットコード  
(sp\_moveと同じ)

spr(3) : パターン番号

spr(4) : 2個目のスプライトのブロッ  
クの中の相対x座標

:

以下2個目、3個目.....と続くという書

## リスト2 EXSPR.FNC

```
1: .INCLUDE      DOSCALL.MAC
2: .INCLUDE      IOCALL.MAC
3: .INCLUDE      FDEF.H
4: =====
5: *スプライト定義と複数スプライト同時表示BASIC関数
6: *EXSPR.FNC
7: =====
8: .EVEN
9:   DC.L      _RTS,_RTS,_RTS,_RTS      *インフォメーションテーブル
10:   DC.L      _RTS,_RTS,_RTS,_RTS
11:   DC.L      TOKEN_TBL
12:   DC.L      PARAM_TBL
13:   DC.L      EXEC_TBL
14:   DC.L      0,0,0,0
15:
16:   TOKEN_TBL: DC.B      'grsp_def',0      *外部関数名テーブル
17:   DC.B      'sp_brock',0,0
18:   .EVEN
19:   PARAM_TBL: DC.L      PAR_GRSP_DEF      *パラメータテーブル
20:   DC.L      PAR_SP_BROCK
21:
22:   PAR_GRSP_DEF: DC.W      int_val      *パラメータIDテーブル
23:   DC.W      int_val
24:   DC.W      char_val
25:   DC.W      char_val
26:   DC.W      char_val
27:   DC.W      void_ret
28:
29:   PAR_SP_BROCK: DC.W      char_val
30:   DC.W      int_val
31:   DC.W      int_val
32:   DC.W      aryl
33:   DC.W      int_val
34:   DC.W      char_val
35:   DC.W      char_val
36:   DC.W      void_ret
37:
38:   EXEC_TBL: DC.L      grsp_def      *実行アドレス
39:   DC.L      sp_brock
40:   .TEXT
41: =====
42: *6万色、256色、16色GR ⇒ SP定義
43: *GRSP_DEF(X,Y,横ブロック数,縦ブロック数,転送開始SPパターン番号)
44: =====
45: grsp_def: MOVE.L      22(SP),D0
46:   LSL.L      #8,D0
47:   LSL.L      #2,D0
48:   MOVEA.L      12(SP),A2
49:   ADDA.L      A2,A2
50:   ADDA.L      D0,A2
51:   ADDA.L      #5C0000,A2
52:
53:   MOVE.L      52(SP),D0
54:   LSL.L      #7,D0
55:   MOVEA.L      #EB8000,A1
56:   ADDA.L      D0,A1
57:
58:   MOVE.W      34(SP),D5
59:   ADD.W      D5,D5
60:   SUBQ.W      #1,D5
61:   MOVE.W      44(SP),D1
62:   SUBQ.W      #1,D1
63:
64:   CLR.L      -(SP)
65:   DOS      _SUPER
66:   MOVE.L      D0,(SP)
67:
68:   LOOP_GRSP_0: MOVEA.L      A2,A0
69:   MOVE.W      D5,D2
70:   LOOP_GRSP_1: MOVEQ.L      #15,D3
71:   LOOP_GRSP_2: MOVEQ.L      #1,D4
72:   LOOP_GRSP_3: MOVE.W      (A0)+,D0
73:   ANDI.W      #5000F,D0
74:   LSL.W      #4,D0
```

```
75:
76:   MOVE.W      (A0)+,D6
77:   ANDI.W      #5000F,D6
78:   OR.W      D6,D0
79:   LSL.W      #4,D0
80:
81:   MOVE.W      (A0)+,D6
82:   ANDI.W      #5000F,D6
83:   OR.W      D6,D0
84:   LSL.W      #4,D0
85:
86:   MOVE.W      (A0)+,D6
87:   ANDI.W      #5000F,D6
88:   OR.W      D6,D0
89:
90:   MOVE.W      D0,(A1)+
91:   DBRA.W      D4,LOOP_GRSP_3
92:
93:   LEA.L      1008(A0),A0
94:   DBRA.W      D3,LOOP_GRSP_2
95:
96:   SUBA.L      #53FF0,A0
97:   DBRA.W      D2,LOOP_GRSP_1
98:
99:   LEA.L      54000(A2),A2
100:   DBRA.W      D1,LOOP_GRSP_0
101:
102:   DOS      _SUPER
103:   ADDQ.L      #4,SP
104:
105:   MOVEQ.L      #0,D0
106:   RTS
107: =====
108: *複数スプライト同時表示
109: *SP_BROCK(アレン番号,X,Y,配列,オフセット,個数,プライオリティ)
110: =====
111:   .EVEN
112:   sp_brock: MOVE.L      12(SP),D0
113:   LSL.W      #3,D0
114:   ADD.L      #EB0000,D0
115:   MOVE.L      D0,A2
116:
117:   MOVE.W      24(SP),D2
118:   MOVE.W      34(SP),D3
119:
120:   MOVE.L      42(SP),A1
121:   ADD.L      52(SP),A1
122:   ADD.L      #10,A1
123:
124:   MOVE.L      62(SP),D1
125:   SUBQ.L      #1,D1
126:
127:   MOVE.W      74(SP),D4
128:
129:   CLR.L      -(SP)
130:   DOS      _SUPER
131:   MOVE.L      D0,(SP)
132:
133:   LOOP1:   CLR.W      D0
134:   MOVE.B      (A1)+,D0
135:   ADD.W      D2,D0
136:   MOVE.W      D0,(A2)+
137:   CLR.W      D0
138:   MOVE.B      (A1)+,D0
139:   ADD.W      D3,D0
140:   MOVE.W      D0,(A2)+
141:   MOVE.W      (A1)+,(A2)+
142:   MOVE.W      D4,(A2)+
143:
144:   DBRA.W      D1,LOOP1
145:
146:   DOS      _SUPER
147:   ADDQ.L      #4,SP
148:
149:   MOVEQ.L      #0,D0
150:   RTS
```



式になっています。たとえば、図1のようなスプライトを定義したいときには、(スプライトのドット数は16ドットだとすると)横4個、縦3個分の大きさですよね。で、スプライトパターンの24番から定義をしたときには、

```
grsp_def(32,16,4,3,23)
と書けばいいってわけですね。
でもって、表示したいときにはまず、sp_brock()の表示情報を、
spr(0)=0:spr(1)=0:
spr(2)=1:spr(3)=64
```

```
spr(4)=16:spr(5)=0:
spr(6)=1:spr(7)=65
:
```

とセットしてからsp\_brock()を使って、  
sp\_brock(0,0,0,spr,0,12,0)  
としてやればいいわけですね。

このプログラム、非常に汎用性を考えて作られているせいか、特にsp\_brock()のほうの引数かなりややこしいですけど、一度わかってしまえば、スプライト定義も簡単だし、デカキャラもスプライトでグリーングリーン動かせるので、すごく楽しいです

よ。  
サンプルプログラムもありますので、がんばって使いこなしてみてくださいね。



## 3Dで模様替えでいっ

さてさていよいよ今月もラストでございます。大上さんの作品で模様替えPRO-68Kこと、MGP68K.BASです。どうぞっ！

MGP68K.BAS for X680x0

(要X-BASIC, マウス)

鹿児島県 大上幸宏

### リスト3 GRSP DEF.S

```
1: .INCLUDE      DOSCALL.MAC
2: .INCLUDE      IOCALL.MAC
3:
4: .XDEF         _grsp_def
5:
6: .OFFSET 8
7: GRTOSP_1:      DS.L 1
8: GRTOSP_2:      DS.L 1
9: GRTOSP_3:      DS.L 1
10: GRTOSP_4:      DS.L 1
11: GRTOSP_5:      DS.L 1
12: GRTOSP_6:      DS.L 1
13: AUTOSIZE EQU -4
14: *=====
15: *6万色、256色、16色GR => SP定義 C関数
16: *=====
17: .EVEN
18: .TEXT
19: _grsp_def: LINK A6,#AUTOSIZE
20: MOVEM.L D3-D7/A3-A5,-(A7)
21:
22: MOVE.L GRTOSP_2(A6),D0
23: LSL.L #8,D0
24: LSL.L #2,D0
25: MOVEA.L GRTOSP_1(A6),A2
26: ADDA.L A2,A2
27: ADDA.L D0,A2
28: ADDA.L #C00000,A2
29:
30: MOVE.L GRTOSP_5(A6),D0
31: LSL.L #7,D0
32: MOVEA.L #EB8000,A1
33: ADDA.L D0,A1
34:
35: MOVE.W GRTOSP_3+2(A6),D5
36: ADD.W D5,D5
37: SUBQ.W #1,D5
38: MOVE.W GRTOSP_4+2(A6),D1
39: SUBQ.W #1,D1
40:
41: CLR.L -(SP)
42: DOS _SUPER
43: MOVE.L D0,(SP)
```

```
44: LOOP_GRTOSP_0: MOVEA.L A2,A0
45: MOVE.W D5,D2
46: LOOP_GRTOSP_1: MOVEQ.L #15,D3
47: LOOP_GRTOSP_2: MOVEQ.L #1,D4
48: LOOP_GRTOSP_3: MOVE.W (A0)+,D0
49: ANDI.W #000F,D0
50: LSL.W #4,D0
51:
52: MOVE.W (A0)+,D6
53: ANDI.W #000F,D6
54: OR.W D6,D0
55: LSL.W #4,D0
56:
57: MOVE.W (A0)+,D6
58: ANDI.W #000F,D6
59: OR.W D6,D0
60: LSL.W #4,D0
61:
62: MOVE.W (A0)+,D6
63: ANDI.W #000F,D6
64: OR.W D6,D0
65:
66: MOVE.W D0,(A1)+
67: DBRA.W D4,LOOP_GRTOSP_3
68:
69: LEA.L 1008(A0),A0
70: DBRA.W D3,LOOP_GRTOSP_2
71:
72: SUBA.L #3FF0,A0
73: DBRA.W D2,LOOP_GRTOSP_1
74:
75: LEA.L #4000(A2),A2
76: DBRA.W D1,LOOP_GRTOSP_0
77:
78: DOS _SUPER
79: ADDQ.L #4,SP
80:
81: UNLK A6
82:
83: MOVEQ.L #0,D0
84: RTS
85:
86: .END
```

### リスト4 SP BROCK.S

```
1: .INCLUDE      DOSCALL.MAC
2: .INCLUDE      IOCALL.MAC
3:
4: .XDEF         _sp_brock
5:
6: .OFFSET 8
7: SP_B_1:      DS.L 1
8: SP_B_2:      DS.L 1
9: SP_B_3:      DS.L 1
10: SP_B_4:      DS.L 1
11: SP_B_5:      DS.L 1
12: SP_B_6:      DS.L 1
13: SP_B_7:      DS.L 1
14: AUTOSIZE EQU -4
15: RET EQU -4
16: *=====
17: *複数スプライト同時表示 C関数
18: *=====
19: .EVEN
20: .TEXT
21: _sp_brock: LINK A6,#AUTOSIZE
22: MOVEM.L D3-D7/A3-A5,-(SP)
23:
24: MOVE.L SP_B_1(A6),D0
25: LSL.W #3,D0
26: ADD.L #EB0000,D0
27: MOVEA.L D0,A2
28:
29: MOVE.W SP_B_2+2(A6),D2
30: MOVE.W SP_B_3+2(A6),D3
31:
32: MOVE.L SP_B_4(A6),A1
33: ADD.L SP_B_5(A6),A1
```

```
34:
35: MOVE.W SP_B_6+2(A6),D1
36: SUBQ.W #1,D1
37:
38: MOVE.W SP_B_7+2(A6),D4
39:
40: CLR.L -(SP)
41: DOS _SUPER
42: MOVE.L D0,(SP)
43:
44: LOOP1: CLR.W D0
45: MOVE.B (A1)+,D0
46: ADD.W D2,D0
47: MOVE.W D0,(A2)+
48: CLR.W D0
49: MOVE.B (A1)+,D0
50: ADD.W D3,D0
51: MOVE.W D0,(A2)+
52: MOVE.W (A1)+,(A2)+
53: MOVE.W D4,(A2)+
54:
55: DBRA.W D1,LOOP1
56:
57: DOS _SUPER
58: ADDQ.L #4,SP
59:
60: MOVEM.L (SP)+,D3-D7/A3-A5
61: UNLK A6
62:
63: MOVEQ.L #0,D0
64: RTS
65:
66: .END
```

模様替えをするとき、悩みませんか？ そう、いきなり家具を動かしちゃうとちゃんとおさまらなくなるし疲れるし、でも、たとえば縮小サンプルを紙で作って、レイアウトしようとしてもそれぞれの家具の高さがわからなくて部屋がイメージしにくかったり……そんな悩みをイッキに解決してくれるのがこの模様替えPRO-68K, MGP68K.BASなのです。なんと3D表示で立体的に部屋の模様替えをシミュレートできるんですよ。

このプログラムはBASICで書かれていますから、

A>BASIC

でBASICを起動して、リストを入力すればそれでOK。

この模様替えPRO-68Kは立ち上げると、家具の大きさなどを決めるモードになっています。ここで部屋の中にあるテーブルや椅子、ライトやベッド、たんすなどの家具を作りましょう。

このプログラムでは10個の家具のオブジェクトを作ることができます。まず、これが0～9のうちのどのオブジェクトであるかキーボードから入力し、画面上にマウスを使って、一筆描きの要領で家具の上から見た形を描いていってください。マウスは左ボタンを押すと前の点から描かれていた線が固定され、その点から次の線を描いていく……とグラフィックエディタのような操作方法になっていて、画面の描画エリアの一番左上、つまり描画を始めた点が左

クリックされるとその家具の形が登録されます。

家具の形状を登録したあと、家具の色を塗る場所を聞いてくるのでマウスで指定して左クリックしてください。色を塗りたくない場合は右クリックです。

それから次に家具の高さをキーボードから入力します。単位はcmで、適当な大きさを数字で入力してください。

最後に家具の名称をキーボードから入力してやれば1つの家具の登録が完了します。

で、これで家具をいくつか作ったら次は部屋の広さを決めます。家具の登録画面で、Nを入力すると部屋の広さの設定画面になります。今度はマウスを動かすことによって部屋の大きさが変わるようになっているので、左クリックで部屋の広さを決定してください。部屋の具体的な大きさ（何畳分あるかなど）の情報は画面下に数字が表示されます。

さて、ここまで終わるとやっとおまちな家具の配置に入るんですね。設置したい家具が登録されているエリア番号をキーボードで入力してから、マウスで置きたい場所を指して左クリックで設置します。マウスボタンを右クリックすると家具が15度単位で回転します。キャンセルするにはESCキーを押してください。

Dを入力すると画面上に線を引くことができます。これでドアや窓の位置を描いておきましょう。これはただの目印なので立体にはならないですけど。

そして、家具の設定が終わるといよいよ

このプログラムの目玉！ なんとこのプログラムでは部屋と配置した家具を3Dで表示したうえに、グルグルと視点を変えて見ることができるようになっています。家具の設置画面で、Nを入力すると疑似3D表示になります。

画面の右に、

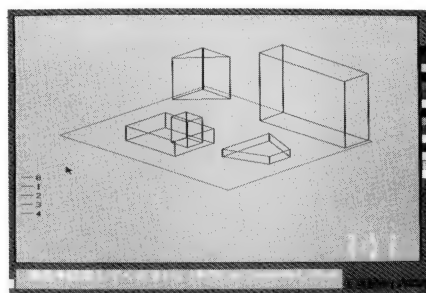
```
5 4 3
6 □ 2
7 0 1
```

と表示されますが、これが部屋をどの位置から見るかの設定になっていて、0から7までのどれかを入れると3Dで見る視点を変えることができます。コマンド画面でLを入力すると、部屋を連続的に回転しながら見ることもできます。なにかキーを押すと終了です。

ふう～。それにしてもこいつはとんでもない力作ですね。コマンドの解説を書くだけでもこんなになっちゃうんだもんね。実をいうと、この解説もひと通り使えるように書いてただけで、実はまだ紹介してないコマンドもあったりする、というこの恐ろしさ。

んーでも、すごーく楽しいんですよ、このプログラム。本当に部屋の模様替えをしようと思うと体力使うけど、部屋をあいうふうにしよう、こういうふうにしよう、って想像するだけでいろいろ楽しめるんですよね。「よーし、部屋は20畳だあ！ トレンディな三角柱のスタンドライトもあるぞ！」なんて財力まで超えてしまったりなんかして(ちょっとむなしい……)。

実はなにかがあるときって、その前の計画している段階が一番楽しいのですよね。実際に遠足に行けばすごい坂があって大変だったり、腹が減ったのに弁当はすべて食べてしまったあとだったりとか。でも想像するだけなら悪いことは全部どこかへいってしまうから、ひたすらに楽しいのですよね。私も子供の頃は遠足の前日はわくわくしてなかなか眠れなかったもんです(そうか、だから遠足や運動会の思い出って「ね



MGP68K.BAS

## リスト5 SAMPLE.BAS

```
10 /* =====
20 /*GRSP_DEF(),SP_BROCK()用サンプルプログラム
30 /* =====
40 screen 1,2,1,1 : img_scrn(0,2,1) : vpage(1)
50 sp_disp(1) : sp_off()
60 dim char spr(511)
70 /*
80 for i=0 to 15
90 palet(i+160,rgb(i*2+1,0,i*2+1)+1)
100 sp_color(i,rgb(i*2+1,i*2+1,i*2+1)+1,1)
110 sp_color(i,rgb(0,0,i*2+1)+1,2)
120 circle(128-i,128-i,(16-i)*4,i+160)
130 paint(128-i,128-i,i+160)
140 next
150 /*
160 for x=0 to 7 : for y=0 to 7 : aa=(y*8+x)*4
170 spr(aa) = x*16 : spr(aa+256)=x*16 /*← x 相対座標
```

```
180 spr(aa+1)=y*16 : spr(aa+257)=y*16 /*← y 相対座標
190 spr(aa+2)=1 : spr(aa+258)=2 /*← パレット
200 spr(aa+3)=y*8+x : spr(aa+259)=y*8+x /*← バターン番号
210 next : next
220 /*
230 grsp_def(64,64,8,8,0)
240 while 1
250 for i=0 to 359
260 xx=cos(i*pi()/180)*64
270 yy=sin(i*pi()/180)*64
280 sp_brock(0,xx+80,yy+80,spr,0,64,3)
290 sp_brock(64,xx*2+80,yy*2+80,spr,256,64,3)
300 if inkey$(0)<>" " then end
310 next
320 endwhile
330 end
```



むい〜、ひたすらねむい〜」って記憶しかないんだな、私は)。

ま、なにはともあれプログラムに大事なのは「想像力」。やっぱり思い込めるってのは大事なんであります。

## リスト6 MGP68K.BAS

```
10 /*-----
20 /* 模様替えPRO68K ver 1.15 -
30 /*
40 /* COPYRIGHT BY Y.OUE 1993 PX9301 -
50 /*-----
60 /** INIT **
70 screen 1,1,1:console 0,32,0
80 int A,B,C,D,P,O,SX,SY,CX,CY,CL,ML,MR,TX,TY,PO,FI,F2,S,W,SP
:float FX,FY,ZX,ZY,XJ,YJ,JJ,SI,CO,RD,SA:str MM,IN,FN,ST[255],CD
90 dim int PX(19,49),PY(19,49),PP(10),PS(10),CR(10),DX(550),D
Y(550)
100 dim float QX(9,49),QY(9,49),LX(3),LY(3),SN(360),CN(360)
110 dim int LS(3)={65278,49087,61423,64507}
120 dim int D1(7)={-1,1,-1,1,-1,1,1}
130 dim int D2(7)={0,1,3,2,1,3,2,0}
140 dim str NM(9)
150 color [0,63488,65472,65535]
160 palette(1,2114):locate 21,15:print"しばらくお待ちください"
170 for I=0 to 9:PX(I,1)=128:PY(I,1)=102:next
180 for I=0 to 360
190 RD=I/(180/pi(1)):SN(I)=sin(RD):CN(I)=cos(RD):I=I+4:next
200 apage(3):vpage(15):PTS(1)
210 box(9,10,491,450,0):fill(494,63,511,302,1)
220 line(8,11,8,449,0):line(492,11,492,449,0)
230 for I=11 to 449
240 line(10,I,490,I,14):next
250 fill(10,462,398,496,14)
260 for I=1 to 15
270 fill(495,48+I*16,511,62+I*16,I):next
280 /** MAIN **
290 /** OBJECT **
300 apage(0):mouse(4):mouse(1)
310 msarea(128,102,378,352)
320 box(123,96,383,358,0)
330 for J=0 to 1
340 for I=1 to 26
350 line(123+(J*256),I*10+92,126+(J*256),I*10+92,1)
360 line(I*10+118,96+(J*257),I*10+118,100+(J*257),1)
370 next:next
380 CX=128:SY=128:CY=102:SY=102:CL=1:O=1:P=1:W=1:SP=0:cls
390 repeat
400 repeat
410 A=0:locate 3,29:print"OBJECT NO.(0-9) or Load/Save/E
dit/Next ?"
420 locate 45,30:if C=0 then print"Edit" else print spc(4)
430 IN=inkey$
440 if IN="L" or IN="l" then LOADOB()
450 if IN="S" or IN="s" then SAVEOB()
460 if IN="N" or IN="n" then B=1:A=1:F1=1:IN="16"
470 if IN="W" or IN="w" then {
480 if W=3 then W=1:SP=0 else W=3:SP=1
490 }
500 locate 40,30:if IN="E" or IN="e" then {
510 if C=1 then C=0 else C=1
520 }
530 if asc(IN)<48 or asc(IN)>57 then F1=0 else EDIT()
540 until F1=1
550 until B=1
560 /** ROOM **
570 fill(118,92,388,362,14):A=0
580 locate 3,29:print"部屋の大きさを指定して下さい":print spc
(16)
590 CL=12:CX=110:CY=100:msarea(12,12,250,230)
600 repeat
610 box(CX,CY,250+(ZX/2),230+(ZY/2),14)
620 mspos(CX,CY):msstat(TX,TY,ML,MR)
630 ZX=abs(250-CX)*2:ZY=abs(230-CY)*2
640 box(CX,CY,250+(ZX/2),230+(ZY/2),12)
650 JX=ZX/180:JY=ZY/180:JJ=(ZX*ZY)/16200
660 locate 3,30:print using"縦"##cm:ZY;
670 print using"##.##度分";JY;
680 print using"横"##cm:ZX;
690 print using"##.##度分";JX;
700 print using"##.##度";JJ
710 if ML=-1 then {
720 X1=CX:Y1=CY:X2=ZX+CX:Y2=ZY+CY:A=1 }
730 until A=1
740 /** PUT **
750 repeat
760 apage(0):wipe():apage(1):wipe():cls
770 msarea(10,11,490,449):window(10,11,490,449)
780 box(X1,Y1,X2,Y2,12):apage(0)
790 repeat
800 locate 3,29:print"家具などを設置してください (0-9) or
Door/Next":SA=0
810 IN=inkey$
820 if asc(IN)<48 or asc(IN)>57 then F1=0 else PUTOB()
830 if IN="D" or IN="d" then DOOR()
840 if IN="N" or IN="n" then F1=1
850 until F1=1
860 /** 簡易 3 D 表示 **
870 apage(1):wipe()
880 for I=0 to 9
890 line(17,(I*16)+298,30,(I*16)+298,CR(I))
900 symbol(35,(I*16)+293,NM(I),1,1,0,1,0)
910 next
920 apage(0):wipe():A=0:B=0
930 repeat
```

それにしても本当にわずか350行のプログラムなのに3D表示まであるってのはすごすぎるっすよね。実行中は使えるコマンドが画面に表示されますから、それを見ながらいろいろ試してみてくださいね。

さて、こんなところで今月はおしまい。ところで、最近BASICのゲームの投稿が少ないんですよー。求むショートなゲーム！できれば思い込みたっぷりなゲームね。では、また来月！

```
940 locate 51,25:print"5 4 3"
950 locate 51,26:print"6 0 2"
960 locate 51,27:print"7 0 1"
970 repeat
980 locate 3,29:print"視点を設定してください。(0-7) or L
oop/Ret./End"
990 IN=inkey$
1000 if asc(IN)<48 or asc(IN)>55 then F1=0 else F1=1:F2=0
:SA=(val(IN)*45):F2=3
1010 if IN="L" or IN="l" then F1=1:F2=1:SA=0
1020 if IN="R" or IN="r" then F1=1:F2=2
1030 if IN="E" or IN="e" then mouse(0):mouse(2):end
1040 until F1=1
1050 cls
1060 repeat
1070 apage(0):wipe():IN=inkey$(0)
1080 SI=SN(SA):CO=CN(SA):FX=ZX/2:FY=ZY/2
1090 if F2=1 then { SA=SA+10-(SP*5)
1100 if SA=360 then SA=0 }
1110 if F2=2 then A=1:F2=3:continue
1120 if IN<>" " then F2=3
1130 for I=0 to 3
1140 LX(I)=X1+((D1(I)*FX*CO)-(D1(I+4)*FY*SI))/1.25+FX:
LY(I)=Y1+((D1(I)*FX*SI)+(D1(I+4)*FY*CO))/2.5+FY
1150 next
1160 for I=0 to 3
1170 line(LX(D2(I)),LY(D2(I)),LX(D2(I+4)),LY(D2(I+4)),1
2)
1180 next
1190 O=0
1200 repeat
1210 if PY(O,49)=1 then {
1220 for I=1 to PS(O)
1230 QX(O,I)=((PX(O+10,I)-FX)*CO)-((PY(O+10,I)-FY)
*SI))/1.25+FX:QY(O,I)=((PX(O+10,I)-FX)*SI)+((PY(O+10,I)-FY)*CO
)/2.5+FY
1240 next
1250 for I=1 to PS(O)-1
1260 line(QX(O,I)+X1,QY(O,I)+Y1,QX(O,I+1)+X1,QY(O,I
+1)+Y1,CR(O))
1270 line(QX(O,I)+X1,QY(O,I)+Y1-(PX(O,49)/1.5*),QX(
O,I+1)+X1,QY(O,I+1)-((PX(O,49)/1.5*),CR(O))
1280 line(QX(O,I)+X1,QY(O,I)+Y1,QX(O,I)+X1,QY(O,I)+
Y1-(PX(O,49)/1.5*),CR(O))
1290 next
1300 }
1310 O=O+1
1320 until O=10
1330 until F2=3
1340 repeat:until inkey$(0)=""
1350 until A=1
1360 for I=0 to 9:PY(I,49)=0:next
1370 until B=1
1380 end
1390 /** COLOR **
1400 func COLORC()
1410 for I=0 to 2000*W:next:apage(3)
1420 msarea(494,63,511,302)
1430 repeat
1440 mspos(CX,CY):msstat(TX,TY,ML,MR)
1450 if ML=-1 then {
1460 CL=point(CX,CY)
1470 fill(494,330,511,345,CL) }
1480 until MR=-1
1490 apage(0):return():endfunc
1500 /** LOAD **
1510 func LOADOB()
1520 locate 3,29:print"ファイル名":print spc(36)
1530 error off:locate 15,29:input FN
1540 D=fopen(FN+".MGP","R")
1550 if D=-1 then {
1560 locate 3,29:print FN+".MGP"は存在しません"
1570 for I=0 to 10000*W:next
1580 error on:IN="":return() }
1590 locate 3,29:print"データ読み込み中です。しばらくお待ちくだ
さい"
1600 fread(DX,550,D):fread(DY,550,D):fread(PP,10,D)
1610 fread(PS,10,D):fread(CR,10,D):fread(ST,D)
1620 for I=0 to 10
1630 for J=0 to 49
1640 PX(I,J)=DX((I*50)+J):PY(I,J)=DY((I*50)+J)
1650 next:next:A=2:B=0
1660 for I=0 to 9
1670 repeat
1680 IN=mid$(ST,A,1)
1690 if asc(IN)=160 then {
1700 NM(I)=CD:CD="":A=A+1:B=1
1710 } else {
1720 CD=CD+IN:A=A+1 }
1730 until B=1
1740 B=0:next:A=0:B=0:C=1
1750 return():endfunc
1760 /** SAVE **
1770 func SAVEOB()
1780 locate 3,29:print"ファイル名":print spc(36)
1790 error off:locate 15,29:input FN
1800 D=fopen(FN+".MGP","R")
1810 if D<>-1 then {
```

```

1820 locate 3,29:print FN+".MGP"+"に上書きしますか? (Y/N)":A
=0
1830 repeat
1840   IN=inkey$
1850   if IN="Y" or IN="y" then A=1:B=1
1860   if IN="N" or IN="n" then A=1:B=0
1870   IN=""
1880 until A=1
1890 if B=0 then return()
1900 locate 3,29:print FN+".MGP"+" 上書きします。";:print sp
c(10) } else {
1910   locate 3,29:print FN+".MGP"+" 新規作成します。" }
1920 fclose(D)
1930 D=fopen(FN+".MGP","C")
1940 for I=0 to 10
1950   for J=0 to 49
1960     DX((I*50)+J)=PX(I,J):DY((I*50)+J)=PY(I,J)
1970 next J
1980 fwrite(DX,550,D):fwrite(DY,550,D):fwrite(PP,10,D)
1990 fwrite(PS,10,D):fwrite(CR,10,D):ST=""
2000 for I=0 to 9
2010   ST=ST+chr$(160)+NM(I):next
2020 ST=ST+chr$(160):fwrite(ST,D):fclose(D):C=1
2030 return():endfunc
2040 /** EDIT **
2050 func EDIT()
2060   O=val(IN):P=2
2070   locate 3,29:print spc(42)
2080   locate 27,30:print spc(13)
2090   apage(1):fill(128,102,378,352,0)
2100   apage(0):fill(128,102,378,352,14)
2110   if C=1 and O<16 then {
2120     for I=1 to PS(O)-1
2130       line(PX(O,I),PY(O,I),PX(O,I+1),PY(O,I+1),CR(O))
2140     next
2150     if PP(O)=1 then paint(PX(O,48)+128,PY(O,48)+102,CR(O))
2160     A=1 }
2170 repeat
2180   msstat(TX,TX,ML,MR)
2190   if C=0 then line(SX,SY,CX,CY,0)
2200   mspos(CX,CY)
2210   if C=0 then line(SX,SY,CX,CY,CL)
2220   locate 3,30:print using"X ###cm";CX-128;
2230   print using"Y ###cm";CY-102;
2240   print using"NO. ##";O;
2250   if ML=-1 then {
2260     PX(O,P)=CX:PY(O,P)=CY
2270     apage(1):line(PX(O,P-1),PY(O,P-1),CX,CY,CL)
2280     if CX=128 and CY=102 then {
2290       apage(0):fill(128,102,378,352,0)
2300       SX=128:SY=102:PS(O)=P:CR(O)=CL:P=2:A=1 }
2310     for I=0 to 1000*W:next
2320     SX=CX:SY=CY:P=P+1:apage(0)
2330   }
2340   if MR=-1 then {
2350     line(SX,SY,CX,CY,0)
2360     COLORC(128,102,378,352):CX=SX:CY=SY
2370     for I=0 to 3000*W:next }
2380 until A=1
2390 if O<16 and C=0 then {
2400   locate 3,29:print"ベイントする位置":A=0:apage(1)
2410   repeat
2420     msstat(TX,TY,ML,MR):mspos(CX,CY)
2430     if ML=-1 then {
2440       paint(CX,CY,CL):PO=point(103,82):A=1
2450       if PO=CL then {
2460         paint(103,82,0):PP(O)=0 } else {
2470         PP(O)=1 }
2480     }
2490     if MR=-1 then PP(O)=0:A=1
2500   until A=1
2510   PX(O,48)=CX-128:PY(O,48)=CY-102
2520   locate 3,29:print"エディットした物の高さ";
2530   input IN:PX(O,49)=val(IN):locate 27,29:print spc(20)
2540   for I=0 to 1000*W:next
2550   locate 3,29:print"エディットした物の名称";
2560   input IN:NM(O)=IN:apage(0) } else {
2570   if O<16 then locate 27,30:print NM(O)
2580 }
2590 return():endfunc
2600 /** PUTOB **
2610 func PUTOB()
2620   O=val(IN):A=0
2630   locate 3,29:print spc(45)
2640   locate 27,30:print NM(O):print spc(10)
2650   if PY(O,49)=1 then {
2660     apage(0):for I=1 to PS(O)-1
2670       line(PX(O+10,I)+X1,PY(O+10,I)+Y1,PX(O+10,I+1)+X1,PY(
O+10,I+1)+Y1,0)
2680     next

```

```

2690     if PP(O)=1 then paint(PX(O+10,48),PY(O+10,48),0)
2700   }
2710 apage(2)
2720 repeat
2730   for I=1 to PS(O)
2740     line(CX+QX(O,I)-S,CY+QY(O,I)-S,CX+QX(O,I+1)-S,CY+QY(O,
I+1)-S,14)
2750   next
2760   mspos(CX,CY):msstat(TX,TY,ML,MR)
2770   SI=SN(SA):CO=CN(SA)
2780   for I=1 to PS(O)
2790     QX(O,I)=(PX(O,I)-128)*CO-(PY(O,I)-102)*SI
2800     QY(O,I)=(PY(O,I)-128)*SI+(PX(O,I)-102)*CO
2810   next
2820   for I=1 to PS(O)
2830     line(CX+QX(O,I)-S,CY+QY(O,I)-S,CX+QX(O,I+1)-S,CY+QY(O,
I+1)-S,CR(O))
2840   next
2850   if MR=-1 then {
2860     if SA=345 then SA=0 else SA=SA+15
2870   }
2880   locate 3,30:print using"X ###";CX-X1-S;
2890   print using"Y ###";CY-Y1-S:print using" A ###";SA:pri
nt using"NO. ##";O
2900   IN=inkey$(0)
2910   if IN=" " then {
2920     apage(0):pset(CX,CY,5):apage(2) }
2930   if IN="S" or IN="s" then {
2940     if S=0 then S=10 else S=0
2950     wipe() }
2960   if asc(IN)=27 then wipe():A=1:PY(O,49)=0
2970   if ML=-1 then {
2980     for I=1 to PS(O)
2990       PX(O+10,I)=QX(O,I)+CX-X1-S
3000       PY(O+10,I)=QY(O,I)+CY-Y1-S
3010     next
3020     PY(O,49)=1:wipe():apage(0)
3030     for I=1 to PS(O)
3040       line(CX+QX(O,I)-S,CY+QY(O,I)-S,CX+QX(O,I+1)-S,CY+QY(
O,I+1)-S,CR(O))
3050     next
3060     PX(O+10,48)=((PX(O,48))*CO-(PY(O,48))*SI)+CX-S
3070     PY(O+10,48)=((PY(O,48))*SI+(PX(O,48))*CO)+CY-S
3080     if PP(O)=1 then paint(PX(O+10,48),PY(O+10,48),CR(O))
3090     A=1 }
3100 until A=1
3110 F1=0:S=0:SA=0:return():endfunc
3120 /** DOOR **
3130 func DOOR()
3140   locate 3,29:print"始点";:print spc(41):apage(2):A=0
3150   repeat
3160     msstat(TX,TY,ML,MR)
3170     if F2=1 then line(SX,SY,CX,CY,0)
3180     mspos(CX,CY)
3190     locate 3,30:print using"X ###";CX-X1;
3200     print using"Y ###";CY-Y1;
3210     if F2=1 then print using" W ###";sqr(pow((CX-SX),2)+pow(
(CY-SY),2))
3220     if F2=1 then line(SX,SY,CX,CY,3)
3230     if ML=-1 then {
3240       if F2=0 then {
3250         SX=CX:SY=CY:F2=1
3260         locate 3,29:print"終点"
3270         for I=0 to 1000*W:next } else {
3280         line(SX,SY,CX,CY,0)
3290         apage(0):line(SX,SY,CX,CY,3)
3300         A=1 }
3310     }
3320     if MR=-1 then {
3330       if F2=1 then {
3340         locate 3,29:print"始点"
3350         line(SX,SY,CX,CY,0):F2=0 } else {
3360         A=1 }
3370       for I=0 to 1000*W:next
3380     }
3390 until A=1
3400 F2=0:return():endfunc
3410 /** 画面の装飾 **
3420 func PTS()
3430   for I=0 to 63
3440     for J=0 to 3
3450       line(0,8*I+(J*2),511,8*I+(J*2),14,LS(J))
3460   next J
3470 MM="機種替えPRO68K"
3480 for I=0 to 2
3490   for J=0 to 2
3500     symbol(400+I,480+J,MM,1,1,1,0,0)
3510   next J
3520 symbol(401,481,MM,1,1,1,9,0)
3530 return():endfunc

```

## ぱーていハンズ

だーっ、8月号でパレットアニメーションの解説を書いたらば、「ねーねー、パレットってどういうものなの?」という質問を受けてしまったんですねー。うーん、パレットっていうのは……だーっ、そこまで説明する必要があ

ったのか?

ってえことで今月は「ぱーていハンズ」と「動かないよと思う前に」をごっちゃにしてパレットについての解説をしちゃいます。さすがに今回は誰でもわかると思うぞ。

## パレット機能ってなんだ?

さてさて、我らがX68000くんなんでありませうが、こいつは知ってのとおりパソコンのなかで



もグラフィックまわりの機能が抜群に充実しているパソコンなんでありますね。いま、「笑って、お仕事」なんていってる(笑って仕事ができるか！ そんなお仕事があったら紹介してほしいぞ、私は)あのパソコンじゃあ、スプライトはないし、テキスト画面に絵は描けないし、「コッTON」だってできないではありませんか。ねえ。

で、その絵を描く画面、グラフィックやテキストなんであります。実はX68000くん、絵や字を描いているときって画面に直に色をつけているわけじゃないんです。

それではどうやっているかっていうと図1を見ていただきます。

X68000くんは絵を描くときに色と画面の間にパレットというものを置いて、ここに色を入れてから画面にベタベタと絵を描いているんですね。このパレットは場合によって16色分だったり256色分のかいパレット(実際人間が持ったら腕がつるだろうな)だったりします。前に描いたのと同じ色でなにか描きたいな、と思ったときには、パレットの同じ位置の色を使って塗ればいいわけです。

で、このコンピュータのパレットというのは、現実のパレット(紙に絵を描くとき使うプラスチックとかでできたあのパレットのほうね)とちょっと違っていることがあるんです。

現実のパレットだったら、パレット上の絵の具を洗い流してまた違う色をのせて絵を描けば、紙の上の前に描いた部分の色は変わらないで、新しく描いた部分は新しい色で描かれます。ところがコンピュータのパレットの場合にはなんと！ 前に描いた部分の色も新しくパレットに出した色に変わってしまうんですね。これがコンピュータ界のマカフシギ。

図1でいうとパレットの下の部分にある色が、画面上の髪とまゆげ(のつもり……なに？ 見えない?)の部分に対応しています。パレット上のその色を変えると、髪とまゆげの色が変わります。さらに、これから画面上にヒゲを描くとすると、ヒゲと髪とまゆげが同じ色になるんですね。

このパレットはX68000くんではグラフィック画面用とテキスト画面用の2つがあってそれぞれの画面の色を変えることができます。

X-BASICの場合、パレットの色を割り当てるには、テキスト画面ならば、

```
color[]
```

という命令で、グラフィック画面は、

```
palet()
```

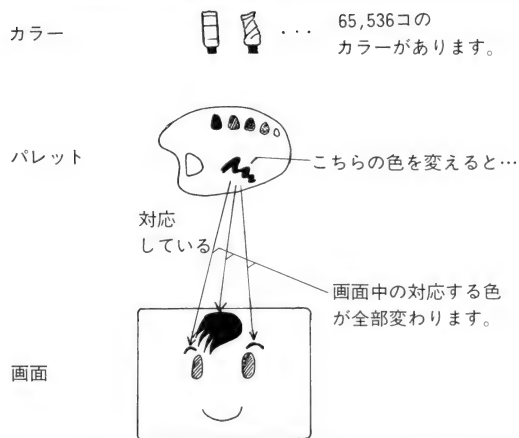
という関数で変えることができます。

実際にサンプルプログラムを作ったので見てみましょうね。このプログラムは15色で線

## リスト

```
5 /* 画面の初期化 */
10 screen 1,1,1,1
15 /* 1色ずつ色を変えながら線を書いて円を作ります */
20 for a=1 to 255
30 dx=cos(pi()*2*a/256)*256
40 dy=sin(pi()*2*a/256)*256
50 line(255,255,255+dx,255+dy,a mod 15 +1)
60 next
65 /* ここからパレットを使います */
70 while(1)
80   for i=1 to 15
90     for j=1 to 15 /* ひとつずつ色をずらします */
100      ncol=((i+j) mod 15) + 1
110      r = (ncol and 2)/2 * 31 /* 次の色を計算します */
120      g = (ncol and 4)/4 * 31
130      b = (ncol and 1) * 31
140      palet(j,rgb(r,g,b)) /* パレットをきりかえます */
150     next
160   next
170 endwhile
```

図1



を放射状に引いて、それぞれの線の色を順番に変えることで、ぐるぐる回転しているように見せるデモです。

このサンプルプログラムでは画面は16色モードを使っています。「screen 1,1,1,1」の2つ目のパラメータの「1」がモードを設定しています。

この16色モードというのは図1でいうと、パレットのところに「16色分の色しかのせられない」小さなパレットを使うと思えばいい合っています。

で、この16色のパレット上の色で1本ずつ線を書いて円を描きます。円は、中心が画面の真ん中(256,256)の位置で、cos(), sin()を使えば円周上の点の位置が求められるので、この2点をとって線を書いていきます。

そして、回転。

いま、それぞれの線に1~15のパレット番号が順番に割り振ってあります。1~15のパレットの色をその前の番号のパレットに入っている色に変えていくと……回転して見えるというわけですね。前の番号のパレットの色はrgb()という関数で求められます。rgb()関数に関してはユーザーズリファレンスを参照してください。

プログラムの実行を中断するときはbreakキーを押してください。

そうそう、このパレットの色を変えるとときに80, 90行で15色使ってますよね。これって実はグラフィックが初期化されたときに、グラフィック画面の背景がパレット番号0番の色で塗られているからなんですね。したがって、ここで0番のパレットの色を変えると背景の色までバシバシと変わってしまうんです。もし、「パレットを変えると画面のなかの色がいつべんに

変わるんだよ」というのを実感したいときにはそうプログラムを書き換えてみてくださいね。

プログラムの実行を中断したときに、グラフィック画面の背景に色がついてテキスト画面の文字が見えにくくなることがあります。そのときには「width 64」などと打つとグラフィックの表示が止まって見やすくなりますよん。

## これから調べるときに

パレット機能に限らず、プログラムを作りたい作りたい作りたい、わいはゲームを作りたいんじゃーっ！ と叫んでみても、はて、なにからすればいいものやらっていうのはよくあることなんですよ。考えてみればX68000くんってカタログを見るだけでも、とてもたくさんの機能を持ってますもんね。表示画面はグラフィックとテキストがあって、128個のスプライト/画面、実画面スクロール、プライオリティ機能、パレット機能、半透明機能、サウンドまわりはFM音源2ch/8オクターブ、ADPCM……。いったいどの機能を使えばいいやら、そもそもこれはいったいどんな機能なんだーっ！ というのもあったりするわけです。

そういうときの解決法。なんてったって、やっぱりそれはマニュアルが一番です。プログラミングはマニュアルに始まりマニュアルに終わる。冗談でも誇張でもなく私はそういう切っちゃいます(ときどきはマニュアルを見ないで人に聞いちゃうこともあるけど、悪い例ですから見習ってはいけません)。

んで、持っているマニュアルをかたっぱしから50音順索引で調べてみましょう。X68000くんの標準では取扱説明書とBASICとワープロのマニュアルくらいしかないからそう時間もかからないはず。ちなみにパレット機能の場合だと、本体に付属のX-BASIC ver2.0ユーザーズリファレンスの44ページから解説があります。

ついでにいうとX-BASIC ver2.0ユーザーズリファレンスの「応用編」にはこのパレットやグラフィック機能、FM音源などX68000くんの持つ便利な機能の使い方がサンプルプログラムつきで出ています。特にゲームを作りたい！ という初心者の方はどの機能を使うかわかるようになるので、目を通しておくとあとで楽ですよん。

さあ、みななもの、がんばるのだ！ そしてショートプロに投稿するのだ。待ってますからね。

# ネストール(Nestor)

Takayama Tadanobu 高山 忠信

ギリシャ神話に登場する人物がタイトルの由来(?)である「ネストール」を発表します。遊び方は「上海」に似ていて、場から同じ数字を2枚セットで取り除いていくというもの。成功率は結構高めですので、気分転換に軽い気持ちで遊ぶといいでしょう。



## 入力方法

リストは、X-BASICで記述されていますのでX-BASICを立ち上げて入力するなり、エディタから入力するなり、自分のやりやすいように入力してください。

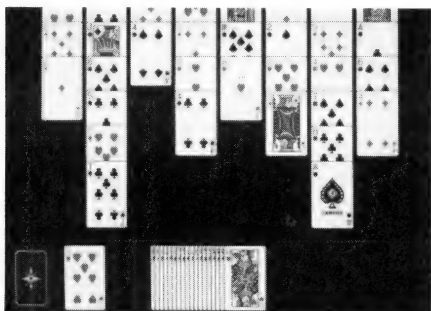
入力が終わったら、コマンドライン上から、

```
A>CARDDRV TR.DAT
```

としてカードドライバを常駐させます。そして、X-BASICのコンフィグファイルに、

```
FUNC=CARD2
```

の1行を追加し、CARD2.FNCを同じディ



レクトリにコピーしておきます。そして、再びX-BASICを立ち上げてから実行してください。

もちろんこれらの組み込み作業は、リストの入力前に行ってもかまいません。

## 遊び方

プログラムを起動すると、8列6段に場札が並べられます。ルール自体は簡単で、重なったカードがないものから同位札を取り除いていくだけです。

もし取れなくなったら左下に伏せてある手札を使ってください。最終目的は場札を全部なくすことです。

もっと砕けた説明をすると「ルールは上海と同じ」といえます。場にあるカードの、いちばん上になっているカードどうしの数字が同じものを右クリックしてください。

1枚目をクリックしたときに、そのカードはマーク(赤い破線で囲まれる)され、2枚目をクリックしたときにその2枚目のカ

ードの数字が同じであれば右下にカードが捨てられます。そして、場にカードがなくなれば成功となります。

## 名前の由来

正直いってよくわかりません。しかし、百科事典から英英辞典まで調べた結果、ギリシャの仲直らせ将軍がその由来だと判断しました。僕もギリシャ神話には強く興味を持っていたので、その人の行動はなんとなく知っていましたが、名前はさっぱりでした。だから、つづりも自信がありません。間違いがわかったならば、変更します、ご指摘をお願いします。一応、彼の名はこう書くのですが、本当にその人物が由来かどうかかわからないので……。

ところで、最近あまり掲載されることがなくなってしまったカードゲームですが、10月号の付録ディスクにも再び収録されたことすし、また、いろいろな作品が生まれるといいですね。

## リスト1

```
10 /*
20 /* Nestor
30 /* Programmed by 高山 忠信 '91.10.17(Thu.)-10.19(Sat.)
40 /*
50 int mx,my,bl,br,count,sunum,chk,tenum,fin
60 dim int card(51),tecd(4),pp(1)
70 prep()
80 repeat
90   init()
100  repeat
110   man()
120   until chk=48 or count=52
130   replay()
140 until fin
150 screen 1,1,1,1
160 mouse(0)
170 end
180 /* 1ゲーム毎の初期化
190 func init()
200   int i
210   apage(1)
220   wipe()
230   for i=0 to 51
240     card(i)=i+1
250   next
260   count=47
270   sunum=0
280   tenum=0
```

```
290   chk=0
300   pp(0)=54
310   shuffle()
320   c_put(14,399,0)
330   coset()
340   for i=0 to 47
350     baset(i)
360   next
370   mouse(1)
380 endfunc
390 /* シャッフル
400 func shuffle()
410   int i,j,k,m,s,t,choo
420   dim char chw(9)
430   mouse(2)
440   for i=0 to 99
450     s=rnd()*52:t=rnd()*52
460     k=card(s):card(s)=card(t):card(t)=k
470   next
480   for i=0 to 7
490     for j=1 to 5
500       check(i,j)
510     next
520     if i=6 then {
530       for j=42 to 50
540         for k=j+1 to 51
550           if number(j,k) and chw(k-42)=0 then {
560             chco=chco+1
```



```

570         chw(k-42)=1
580     }
590     next
600     next
610 }
620 if chco>4 then shuffle():break
630 next
640 endfunc
650 /* 指定の列の重複をチェック
660 func check(a,b)
670     int i
680     for i=0 to b-1
690         if number(a*6+b,a*6+i) then {
700             move(a*6+b)
710             check(a,b)
720         }
730     next
740 endfunc
750 /* 指定したカードを最後に移動
760 func move(a)
770     int i,s
780     s=card(a)
790     for i=a to 50
800         card(i)=card(i+1)
810     next
820     card(51)=s
830 endfunc
840 /* プレイヤーの処理
850 func man()
860     int k
870     repeat
880         msstat(mx,my,bl,br)
890         mspos(mx,my)
900         until bl or br
910         k=-(pp(0)<54)
920         pp(k)=select()
930         if k=0 then {
940             if pp(0)<52 then p_box(pp(0),5)
950             if pp(0)=52 then treset():pp(0)=54
960         }
970         if k=1 then {
980             p_box(pp(0),0)
990             if pp(1)<52 and pp(0)<>pp(1) then {
1000                 if number(pp(0),pp(1)) then {
1010                     toru(pp(0)):toru(pp(1))
1020                 } else m_play(2)
1030             }
1040             pp(0)=54
1050         }
1060         if pp(k)=53 and count=51 then count=52
1070         if pp(k)=54 then pp(0)=54
1080         repeat:msstat(mx,my,bl,br):until bl=0 and br=0
1090     endfunc
1100 /* プレイヤーの指すカード
1110 func select()
1120     int i,x,y,k
1130     if (mx>50 and mx<463) and (my>16 and my<374) then {
1140         x=(mx-51)*52:y=5
1150         for i=0 to 5
1160             if card(x*6+5-i)=0 then y=4-i
1170         next
1180         if y>-1 then {
1190             if (my>y*52+16 and my<y*52+112) then return(x*6+y)
1200         }
1210     }
1220     if (my>398 and my<496) then {
1230         if (mx>68+tenum*8 and mx<116+tenum*8) and tenum>0
then return(tecd(tenum))
1240         if (mx>13 and mx<63) then {
1250             if count=51 then return(52)
1260             if count=51 and (my>434 and my<460) then return(
53)
1270         }
1280     }
1290     m_play(2):return(54)
1300 endfunc
1310 /* プレイヤーの指すカードを示す
1320 func p_box(a,b)
1330     int x,y
1340     apage(0)
1350     if a<48 then {
1360         x=(a*6)*52+50:y=(a mod 6)*52+15
1370         box(x,y,x+49,y+98,b,&HEEEE)
1380     } else {
1390         box(68+tenum*8,398,118+tenum*8,496,b,&HEEEE)
1400     }
1410     apage(1)
1420 endfunc
1430 /* カードを取る
1440 func toru(a)
1450     int x,y
1460     if a<48 then {
1470         x=a*6:y=a mod 6
1480         fill(x*52+51,y*52+16,x*52+98,y*52+112,0)
1490         if y>0 then baset(x*6+y-1)
1500         m_play(3)
1510         chk=chk+1
1520     } else {

```

```

1530         fill(68+tenum*8,399,116+tenum*8,494,0)
1540         tecd(tenum)=0:tenum=tenum-1
1550         if tenum>0 then c_put(69+tenum*8,399,card(tecd(ten
um)))
1560     }
1570     suteru(card(a))
1580     card(a)=0
1590 endfunc
1600 /* 同じ数か?
1610 func number(a,b)
1620     a=card(a)-1:b=card(b)-1
1630     return((a-(a*13)*13=b-(b*13)*13))
1640 endfunc
1650 /* 場札の配置
1660 func baset(a)
1670     int x,y
1680     x=(a*6)*52+51:y=(a mod 6)*52+17
1690     c_put(x,y,card(a))
1700     line(x+1,y-1,x+45,y-1,1)
1710 endfunc
1720 /* 手札の配置
1730 func treset()
1740     tenum=tenum+1
1750     count=count+1
1760     tecd(tenum)=count
1770     m_play(1)
1780     line(68+tenum*8,400,68+tenum*8,493,1)
1790     c_put(69+tenum*8,399,card(tecd(tenum)))
1800     coset()
1810 endfunc
1820 /* 手札の枚数の表示
1830 func coset()
1840     fill(34,495,50,511,0)
1850     symbol(34,495,itoa(51-count),1,1,1,15,0)
1860     if count=51 then {
1870         fill(14,399,62,511,0)
1880         fill(14,435,62,459,5)
1890         symbol(14,439,"END",1,1,1,15,0)
1900     }
1910 endfunc
1920 /* 捨て札の配置
1930 func suteru(a)
1940     m_play(1)
1950     line(177+sunum*5,400,177+sunum*5,495,1)
1960     c_put(178+sunum*5,399,a)
1970     sunum=sunum+1
1980 endfunc
1990 /* リプレイ
2000 func replay()
2010     int j,k,x
2020     apage(0)
2030     wipe()
2040     mouse(2)
2050     x=-(chk=48)*6
2060     fill(143+x,195,367-x,315,1)
2070     box(144+x,196,366-x,314,15)
2080     if chk=48 then {
2090         symbol(159,211,"Congratulations!",1,1,2,11,0)
2100     } else {
2110         j=48-chk:k=-(j<10)*12
2120         symbol(165+k,211,itoa(j)+"枚 残りました",1,1,2,15,0)
2130     }
2140     symbol(175,251,"Try again?",1,1,1,15,0)
2150     fill(215,283,255,299,5)
2160     fill(263,283,295,299,5)
2170     symbol(223,283,"Yes",1,1,1,15,0)
2180     symbol(271,283,"No",1,1,1,15,0)
2190     msarea(215,283,295,299)
2200     mouse(1)
2210     repeat
2220         msstat(mx,my,bl,br)
2230         mspos(mx,my)
2240         until bl or br
2250         if mx>262 then fin=-1
2260         wipe()
2270         msarea(0,0,511,511)
2280     endfunc
2290 /* 準備
2300 func prep()
2310     randomize(val(mid$(times$,4,2)+right$(times$,2)))
2320     screen 1,1,1,1
2330     mouse(0):mouse(4)
2340     vpage(0)
2350     console , , 0
2360     apage(2)
2370     fill(0,0,511,511,8)
2380     symbol(39,150,"Nestor",3,4,2,10,0)
2390     box(0,391,158,511,9,&HCCCC)
2400     box(160,391,511,511,9,&HCCCC)
2410     vpage(15)
2420     m_init()
2430     for i=1 to 3
2440         m_alloc(i,100):m_assign(i,i)
2450     next
2460     m_trk(1,"q3@45v11t200o2c4")
2470     m_trk(2,"q8@15v13t100o3c4")
2480     m_trk(3,"q2@52v 9t200o4g8")
2490 endfunc

```

X68000・Z-MUSIC用

## 渚のアデリーヌ

Kato Takashi 加藤 隆

X68000・Z-MUSIC用(SC-55対応)

## エロティカ・セブン

Nakata Kenichi 中田 健一

今月も2曲と少な目ですが、内容は充実のLIVE in。どちらもみなさんおなじみでしょう。クレイダーマンといえばあれ、というほどのポピュラーなあの曲と、サザンオールスターズのバリバリの新曲です。進藤氏のコラムも元気に復活！

## まあいすポンジケーキ？

それはマドレーヌや。今月も順調にボケから始まったLIVE inのコーナー、さっそく1曲目を紹介しましょう。Z-MUSICシステム用に「渚のアデリーヌ」です。タイトルを知らない人でも一度くらいは聞いたことあるはず。「俺は知らないね」って人もとりあえず入力して、本当に知らないかチェックしてみてね。きっと知っていると思いますよ。加藤君の原稿にも「いわゆる『渚のアデリーヌ』です。」とあります。うーん、まったくそのとおりかもしれないですね。

MIDIやPCM8.Xも使用していませんので、本体だけで演奏可能です。ピアノ曲なのでMIDIを持っている人は移植の練習台にするのもいいかもしれません。ともかくX68000とZ-MUSICシステムを持っている

人は入力してね。

作品は、FM音源ながら大健闘しています。多少強引と思えるフシもなきにしもあらずですが、加藤君の解釈という見方をすれば奏者のクセや特徴が出ているとも考えられます。ピアノは奥が深いなあ、うんうん。

加藤君といえば、「スプーンおばさん」(1991年2月号)、「魔法の妖精ペルシャ」(同6月号)、「ショパン」(1992年4月号)と掲載歴があります。常連さん一歩手前といったところでしょうか。今回も含めて考えると、だんだんと曲の好みがピアノ系に移ってきたといえるのかもしれませんがね。あるいは得意なジャンルなのかも。こういった得意なものをもつのは上達への早道かもしれませんね。

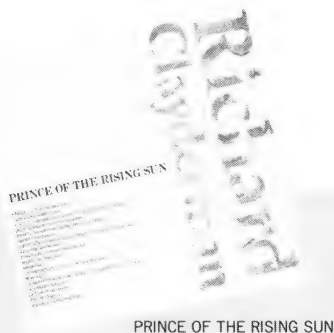
## 喉の渇きをいやすのは

さて、2曲目は、サザンオールスターズの「エロティカ・セブン」です。この原稿を書いている時点ではまだ最新の曲で、有線などでもバリバリに流れています。2枚のシングルの同時発表という無謀とも思われる行為をしながら、2曲ともランクインするあたりはサザンならではのしょうか。

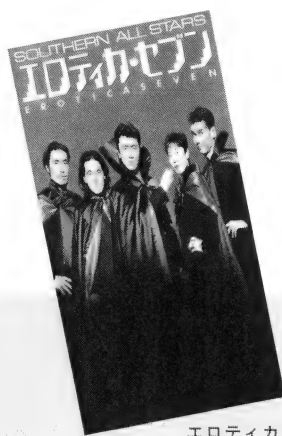
演奏にはSC-55同等品が必要です。また、サンプリングも使っていますので、ミキシングにも注意を払ってください。エフェクトだけにしか使っていないようなので、最悪の場合、なくても聴くことは可能ですが、やっぱりあったほうがよいでしょう。電波新聞社のミキサーケーブルを使う場合は、本体側、MIDI側ともに12時の方向にするとよいそうです。

オリジナルのエフェクトも含めて、よく雰囲気をつかんだ作品ですね。今後の課題は曲をいかに短くまとめるかという点でしょう。ZPPなどを使ってみることをお勧めします。ほかの人に聴いてもらうためにも、入力しやすい作品にするのはテクニックのうちというものです。

サザンといえば、ライバル(?)の松任谷由実が登場していないのは不思議ですね。「真夏の夜の夢」でなくとも、名曲はたくさんあるんですけどねえ。(SIVA)



PRINCE OF THE RISING SUN



エロティカ・セブン

## リスト1 渚のアデリーヌ

1: comment Paul De Senneville 作曲 『渚のアデリーヌ』 by kurkun  
2:  
3:(i)  
4:(v71,0,58,15,2,0,220,0,0,0,0,3,0,28,4,0,15,1,37,2,1,7,0,0,22,9,1,15,1,47,2,12,0,0,0,29,4,3,15,1,37,1,3,0,0,15,7,0,15,10,0,2,1,0,0,1)  
5:(v74,0,57,15,2,0,205,27,0,0,0,3,0,0,5,2,15,2,56,0,3,7,0,1,24,5,2,15,2,53,0,3,7,0,1,24,5,2,15,2,23,0,1,3,0,1,21,6,5,15,2,2,1,0,0,1)  
6:(v79,0,52,15,2,0,205,28,0,0,0,3,0,30,23,1,15,3,21,2,1,2,0,1,19,2,1,15,0,0,3,1

,3,0,1,22,20,11,15,13,27,2,0,3,1,1,31,4,1,15,1,9,1,1,7,0,1)  
7:(v82,0,59,15,2,0,205,30,0,0,0,3,0,24,24,3,15,2,30,2,3,3,0,1,22,11,1,15,0,23,3,1,7,0,1,21,19,17,15,0,34,2,1,0,2,1,23,17,1,15,1,0,3,1,3,0,1)  
8:(v84,0,51,15,2,0,205,0,0,3,0,3,0,29,23,2,4,2,27,2,3,3,0,1,22,15,0,2,0,25,2,1,7,0,1,21,3,0,5,1,27,2,1,3,0,1,19,29,1,5,2,0,3,1,0,0,1)  
9:(m1,3000)  
10:(m2,3000)  
11:(m3,3000)

日本音楽著作権協会(出)承諾第9371652-301号





[illegible]



```

C>B>
235: <C>4>R|:7G>|:R|:3G>|:|:|:3|:4G>|:R|:7G>|:R|:3G>|:|:| <C>4>R|:7G>|:R|:3G>|:|:|
|:|:4G>|:R|:7G>|:R|:3G>|:|:|:|:24G>|:R4C>8R4C<C>8>
236: <L8C>4>|:5RD>|:|>G16G+16A+RG+|:|<|:6RD>|:|>G16G+16A+RG+|:|<|:6RD>|:R|2L>16
237: <C>4>L16R|:7G>|:R|:3G>|:|:|:|:6|:4G>|:R|:7G>|:R|:3G>|:|:| <'C>2'R2 C>4>R|:7G>|:
|R8>C>8R4 C>4R2>|
238: <C>4>R|:7G>|:R|:3G>|:|:|:|:6|:4G>|:R|:7G>|:R|:3G>|:|:|:|:7G>8|: <C>8>|:|:7G>8|: <
C>8>
239: <C>4>R|:7G>|:R|:3G>|:|:|:|:3|:4G>|:R|:7G>|:R|:3G>|:|:| <C>4>R|:7G>|:R|:3G>|:|:|
|:|:4G>|:R|:7G>|:R|:3G>|:|:|:|:4G>|:R4C>8R4R4|
240: L8C>4>|:7G>|:|:7G>|:R4C>C<C>4C>4C>4C> R4C>4R2>L16
241: <C>4>R|:7G>|:R|:3G>|:|:|:|:3|:4G>|:R|:7G>|:R|:3G>|:|:| <C>4>R|:7G>|:R|:3G>|:|:|
|:|:4G>|:R|:7G>|:R|:3G>|:|:|:|:4G>|:R4C>8R4R4|
242: <C>4>R|:7G>|:R|:3G>|:|:3G>|:R|:5G>|:G48G>RG+|:4G>|:R|:7G>|:R|:3G>|:|:|:|:1
6G>|: <C>4>R|:7G>|:R|:3G>|:|:4G>|:R|:5G>|:A>8RG>RA>|:4G>|:R|:7G>|:R|:3G>|:|:|
|:4G>|:G48C>C8C>8C>4> R1 R2.R8C8 R1
243:
244: /CHORUS 2
245: (T12)|:38R1|: @54f6P00U1100V10005L8R4.DDDCD
246: EEEEDD4C>C>B<CDE16F16GRA& AGF<C>GFE D+&D+D+4G+4RE& EEEED4DC>C>B<CDE16F1
6GR& FEDE4DC>B& B2R4C&
247: C1|:|:20R1|: R2R>AGF A2RAB<C D4.C>GCG>G B4B4BF+G+G+ B1 @64U01100V11002(B2>
G)>@54f6P00U1100V10005L8DCC
248: EEEEDD4C>C>B<CDE16F16GRA& AGF<C>GFE D+&D+D+4G+4RE& EEEED4DC>C>B<CDE16F1
6GR& FEDE4DC>B& B2R4C&
249: I10127W>+27>20R1|:3|:R1|:01R8E2...& E1054f6P00U1150V110|:3|R1|:
250: @530f6200300U+1018R405G6'CE'>R'>GB'>AC'>R'>G'>R'>EG'>Q8'DF'>C>A'>Q7'CD'+>CD'+
R4 R4Q6'>CE'>R'>G+H'>Q8'>4B'>Q6'>BD'>C'>A2'>C'>R2'>Q6R4'>CE'>R'>GB'>'>C'A'>R'>E'>R'>8'>E'>
'DF'>'>C'A'>Q6'>CD'+>CD'+R4 R4'E'>B'>R4G'>E'>B'>'>E'>B'>'>E2'>B'R2 R1 R10U-100M
251: @54f6P00U0150W1100|:3|R1:|: @54f6P00U0150W1100R05L8DCC
252: EEEEDD4C>C>B<CDE16F16GRA& AGF<C>GFE D+&D+D+4G+4RE& EEEED4DC>C>B<CDE16F1
6GR& FEDE4DC>B& B2R4C&
253: C1|:|:R1|: @U127V12711012703E1054f6P00U1100V100
254: O5EEEDDDC>C>B<CDE16F16GRA& AGF<C>GFE D+C>B+A&AA+&A& EEEED4DC>C>B<CDE1
6F16GR& FEDE4DC>B& B2R4C&
255: @920U1150W1150M4006L8C1>A2&A+C+G+E'>A1& A1C>A>C>A&E&A <C>A4.&A2 A4.D<D4C
4>A1& A2&AG4.& G1& (G2.)>G'>R06&A1
256:
257: /EFFECT
258: (T25)|:87R1|: 03D1|:4R1|: 03R4.E2&E8& E1|:3|R1|: F2.C4|:42R1|:
259:
260: (P)

```

## リスト4 エロティカ・セブンの音色コンフィグファイル

```
.O3C = WOO.PCM,v95
1 = SCRCH6.PCM,p-11,v200
.O3D = 1,p-11,v115
.O3E = GRSCRSH.PCM,v70
.O3F = VIBSL.PCM,v200,p-2
.ERASE 1
```

## リスト5 エロティカ・セブンのカウンタ表示

1:00006900	00000000	2:00006900	00000000	3:00006900	00000000	4:00006900	00000000
5:00006900	00000000	6:00006900	00000000	7:00006900	00000000	8:00006900	00000000
9:00006900	00000000	10:00006900	00000000	11:00006900	00000000	12:00006900	00000000
25:00006900	00000000						

(進)の  
「ちょっといいですかあ？」

申しわけありません！ 始まっていきなりお休みしてしまうとは……。本当にいつまで続くのか心配になってきたぞ。トホホ。

では気を取り直していってみましょう。

## ★渚のアデリーヌ

OPMでのピアノがいい味を出しています。SC-55などへの移植も楽にできそうです。欲をいえば、クラシックのファジーなテンポと多少の強弱の変化が欲しいところです。

★エロティカ・セブン

これはなかなかの力作です。ネタとしてもタイムリーで、よろしいんじゃないでしょうか。

気になるのはコーラスが強いのと、少々ベタ打ちっぽい点です。ノリはそこそこありますが、音符を並べただけではあまりに機械的。特に工夫が必要だなと感じるのは、ギターとドラム、そしてボーカル。このままだと息つく暇もないので、自分で一度、歌ってみるのもいいかもしれません。SAXアレンジでも同様です。

私はこの曲が主題歌のドラマをビデオに録って欠かさず観ていましたが、あれには続編があ

るんでしょーうかね……。

SC-55のエフェクト設定は

@Enl\_n2

(n1=REVERB depth n2=CHORUS depth)  
 というコマンドを使いますが、それに関しての  
 ちょっとした注意点。

エフェクタのセッティングは楽器にとっては忙しいことですから、コマンドのあとに多少の休符を挟んでやるべきです。曲の出だしで一瞬リバーブやコーラスの効果がまったく感じられないことがあるのは、この休符を忘れてしているからです。曲の途中でエフェクタを切り替えるときはそれほど気になりませんが、先頭でこれやってしまうと、ちょっといただけません。曲の先頭では、ただでさね忙しい楽器のパラメ

タ設定がわんさで行われますから、なおさら注意しないといけないわけです。

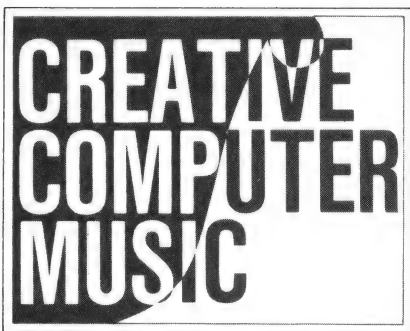
今月の「エロティカ・セブン」。一見正常な演奏に聴こえますが、これをコンパイルして鳴らしてみてください。

ZMUSIC -C EROTICA.ZMS

ZP EROTICA.ZMD

曲の先頭ではエフェクタの効果が表れていません。これを回避するには、すべてのトラックに2分休符程度を入れるといいでしょう。もちろん、MIDIトラックではエフェクト設定のあとに休符を入れないと意味がありません。

先月号に掲載された館野さんの「PASSING BREEZE」ですが、ZPDを作るためには1993年8月号が必要です(スーパーハングオンのPCMデータを分解するため)。気がついた人も多いでしょうが念のため。ところで、この曲のZPDデータは私のOutRunに使ったものとコンパチのようです。名前を変えるだけで対応できてしまう親切設計は嬉しいですね。(准藤麻利)



# Creative Computer Music入門(26)

## 調性の誕生と和音の機能

曲の印象に大きく影響することのひとつが「調性」です。現在は長調と短調がありますが、最近は無調音楽や、頻繁に移調する曲も多いので、原理を知ることが音楽を扱ううえで参考になるでしょう。この調性について、前回説明した和声進行の原理から関連して考えてみます。

Taki Yasushi 瀧 康史

### § 調性の誕生

今月は「調性」について説明します。曲全体または一部がある音を主音とする長調または短調によってまとまりを形づくっている、そのことを調性といいます。「ト長調の曲」とか「移調する」とかいう、あれです。

前回、2つの音の最も基本的な関係は完全5度であり、この完全5度から、全音階と半音階が生まれたと書きました。しかし、今回は全音階については述べましたが、この全音階を実際に使用するのに必要な調性についてはまったく触れませんでした。実際には、全音階を決定しただけでは不十分です。なにしろ、全音階には7つの構成音があります。それぞれ、7つの同じ構成音を使っただけでも7種類の音階ができてしまいますよね。

空間の中でもものを測るときには、基準点と、基準となる長さが必要となります。同じように、音空間の中でも基準が必要です。全音階の決定は、音空間の中の「基準となる長さ」を表します。しかし、空間の中で基準となる長さを決定しただけでは、相対的な関係がわかるだけで、実際の位置などを調べることはできません。音空間においてもこれは同じことです。したがって、基準点、いわばゼロとなる点、すなわち「主音」を定めます。これは、全音階のなかで始まりとなる点を決めることを意味します。

同じ全音階の音で構成される調を考えてみましょう。たとえば、一般的なドレミファソラシド(CDEFGABC)。この調性はすぐにわかるとおり、ハ長調(C major)を表しますが、同じ全音階の構成音からなるものには、ラシドレミファソラ(ABCDEFGA)、つまりイ短調(A minor)もあります。この2つの調は同じ音を使っているにもかかわらず、まったく違った性格をもっています。したがって、調性を決めるということはすなわち、ある全音階のなかで<sup>せんぽう</sup>旋法を選ぶことと主音を決めること、この2つのファクターが必要だということになります。

主音が選ばれるということは、調を決定することになり、主音を起点とするある全音階を決定することは、その旋法を決めることにほかならないのです。

### § 7つの旋法が2つに淘汰されるまで

全音階の構成音は7つですから、各々の音を主音として7つの旋法(≡音階)が生まれることになります(図1)。これらは6世紀ごろからのグレゴリオ聖歌に使われていたため、総称して「グレゴリオ旋法」といいます。

しかし、自然界において首の短いキリンが淘汰されたように、7つの旋法も和声音楽の発展と共に淘汰されて、18世紀の古典期には「長調」といわれる「イオニアモード(長旋法)」と「短調」と呼ばれる「エオリアモード(短旋法)」の2種類(図2)になり、現在に至っています。

こうやって淘汰された結果、2つの旋法が残ったというのはおそらく、ほかの調に比べてこの2つの調が優れた点があったということを表しているのでしょう。

さてその理由なのですが……実はそれを解明することが、今回のテーマなんですね。

### § 古典的24調

原因を究明するために、長調と短調を題材として、和声の構成を調べていきましょう。

それでは「調」について考えてみます。調は「旋法」と「主音」の決定によって成り立つことは、先ほど述べたとおりですね。そこで旋法ですが、これは各々の音の相互関係を意味しているわけで、当然、長調と短調の2つになります。そして、主音を決めることで調が決定されます。スケールは全音階の上に作られますが、半音階を構成する12個のすべての音が主音となり得ますので、すなわち12個の調があります。そして、それぞれ長調と短調の2つの旋法がありますから、その結果、古典的な調性が24個完成されます。

### § 和声進行の原理

音階が完全7度から完成されたように、和声関係を規定する原理は完全5度です。したがって和声進行は7つの固有和音、すなわちダイアトニックトライアドコード



図1 グレゴリオ旋法

(図3)がIを両極にして、完全5度の関係に配列されます。

一定調の固有音関連においては、どうしても1つの減5度は避けることができません。これが和音の進行の最も基本的なことです。

図4を見てください。

まず、5度の進行を下行するものを「D進行」(ドミナントモーション)といいます。それに対して、5度の進行を上行するものを「S進行」(サブドミナントモーション)といいます。

スケールの主音3和音である「I」は常にこれらの進行における基準となります。和声はIから動き、Iに戻るというわけです。この基準点であるIへ最終的に到着する進行であるV-I進行とIV-I進行は、カデンツの最も基本的なものになるわけです。

このIを「トニック(T)」といいます。

ちょっと整理してみましょう。

- ・ I は調の中心点になる
  - ・ 中心点としての I の機能をトニック (T) と呼ぶ
  - ・ T は完全に安定している
- そして、いままでの連載のなかから、
- ・ 安定した和音Tは、どの和音に進行してもかまわない

図2 イオニアモード(長旋法)とエオリアモード(短旋法)

図3 ダイアトニックトライアドコード

と、この4つのことがいえるでしょう。

I以外の和音はすべて不安定であるといえるわけで、音重力の作用を受けて、5度進行をしようします。これが、すべての和声進行の根底に流れるドミナントモーション(D進行)という進行です。セカンダリドミナント(D<sub>2</sub>)という言葉を読載中に何回か使いましたが、これは、ドミナントに対するドミナント、すなわちD<sub>2</sub>からDに進行すること自体がドミナントモーションになります。これを繰り返すとどうなるか。答えは7回目のドミナント、すなわちD<sub>7</sub>の段階でIに戻ってきてしまいます。図5を参照してください。

試しにこの音の進行をZMSデータなどにしてみれば、きっと面白いことがわかりますよ。どこかで聴いたことのある進行だな〜って、そう思うはずです。

さて、ドミナントモーションとはまったく別の進行で、サブドミナントモーション(S進行)という進行があります。これは理論的なものであり、実際には禁じられた進行で「机上の空論」的なものです。ですから、ここでは説明は省略します。

## § 導音の機能

導音というのは、ひと言でいってしまうと、主音の1つ下、すなわち7度の音です。四声体のときに説明しましたが、この音は主音にしか進むことができません。

Vがドミナントモーションをすることは、すなわちトリックIに完全な形で復帰することを表します。これがドミナントモーションの基本進行で、「解決」といいます(図6)。

四声体のときに説明したとおり、音は、理由がない限り最も近くへ進もうとします。したがって、連続度進行(先月号を参照のこと)はこの典型です。このような隣接

音に進みたがる傾向は、音重力に従って下がるときよりも、音重力に反抗して上に行く場合によく表れます。また、このとき、音は全音上よりも半音上を欲します。

このような短2度進行により7度の音は主音に戻るひとつ手前といった意味で、特別な意味をもち始めます。なにしろ音の完全解決のひとつ手前なのですから。よって、和声学ではこの音に特別な意味を考え、これを「導音」と名づけたのです。

和声的に7度の音が導音になるのは、唯一、D(V)の3音のときだけです。この意味はもちろん、V-Iの和声の解決進行を表しているというわけです(図7)

簡単にいうと導音は主音の短2度下なのですが、そう考えてしまうと短調には導音はなくなってしまいます。しかし、私たちはいままでの学習で、長調における導音の導き出すドミナントモーションは非常によい効果をもたらし出すことを知っています。したがって、短調においても、固有の7度の音を半音高めて、人為的に導音を作り出します。すなわち、短調ではドミナントは自然短和音でマイナーコードになるのに、和声的短和音ではドミナントはメジャーコードになります。

ずいぶん前に、短音階でドミナントをメジャーにする理由は、曲があまりにも暗くなってしまうためだと書きましたが、それは結果論で、実はこういった裏づけがあったのです。

自然的短音階において7度の音は短7度であり、強引に長7度にしてしまうと、短音階のニュアンスが崩れてしまいます。そういうことで、短音階のなかで、7度の音を長7度として扱うのは、ドミナントの場合のみです。このことはよく理解して扱わねばなりません(図8)。

## § イオニアとエオリアが残った理由

さて、ひととおり説明したところで、最初の問題である、イオニアモードとエオリアモードが残った理由を説明しましょう。

いままで説明した内容で、だいたい想像がつくと思いますが、イオニアモード(長調)とエオリアモード(短調)は和声進行に適していることがわかります。

そこで、もう少しこれらのスケールの利点を挙げてみましょう。

### 1) イオニアモード(長調)

長調の特質はまず、3つの重要な3音、すなわちI、IV、Vの3つの上いずれも自然3和音(長3和音)が構成されるということです(図9)。そして、いままで述べた、V-I進行に導音進行が含まれるということ(図10)。最後に、Vの上に自然4和音・自然5和音が構成できるということ(図11)です。この性質は、イオニアモードしかもつことができない特色です。

### 2) エオリアモード(短調)

図4 D進行(ドミナントモーション)とS進行(サブドミナントモーション)

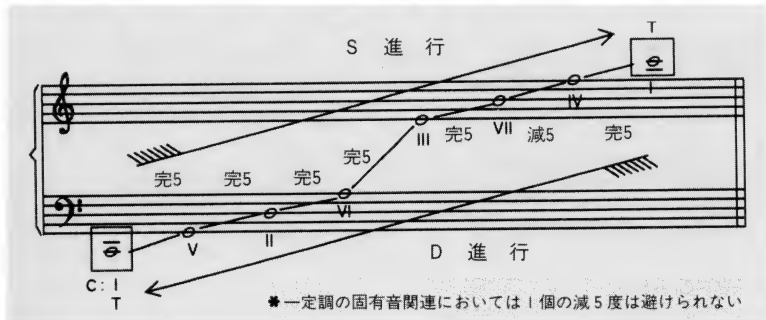


図5 不安定な位置にある和音は5度下の和音を指向する

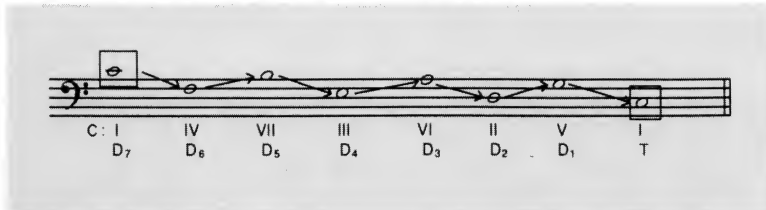




図6 解決



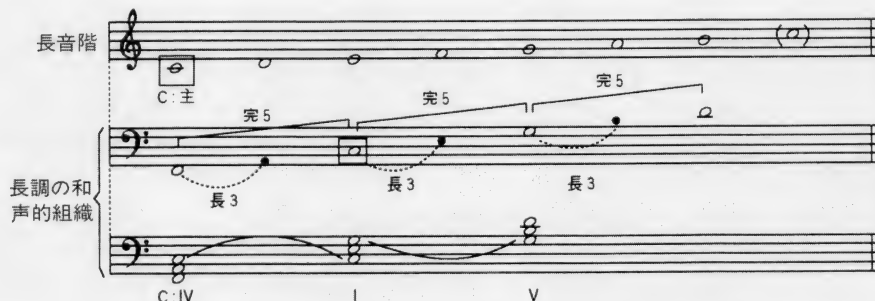
図7 V-Iの和声的解決進行



図8 人為的な7度



図9 長調ではI, IV, Vの上に自然3和音(長3和音)が構成される



長調がそれ自体に和声的調整を十分すぎるほどもっていたのに対して、短調にはそれがありません。ではなぜ、短調が存在価値をもったかといえは、それは長調との微妙な関連によってです。

短調は主要3和音すべてが短3和音です。これはまさに長調の場合と対をなすことになります。この「対であること」が短調の存在価値です。

短調のV-I解決は導音進行を人為的に構成しなければならぬため、先ほど述べたとおり、Vは長3和音にします。このことは短調が独自に調整をもつためには不十分であり、エオリアモード独自の雰囲気を持ててでも長調の特質を借りなくてはならないことを意味します。

## § 最後に

今回は短いですが、とりあえずキリのよいところで終わらしましょう。

ここのところ内容が重いですが、辛抱してください。なにせ原理編というのは私情が入らないので、必要な

ことだけがかなり濃縮されているんですよ。その結果、たった数行に重要なことがぎっしりつまっているため、読むほうも大変かもしれませんね。

そういえば、読者の方からお手紙をいただきました。これはとても励みになるんですが、ちょっと事情がありまして、お返事が書けなくてすいません。よろしかったらまたお手紙ください。

それにしても今回は、私事ですが、クソメモリのせいで苦戦してしまいました。あまりに寂しいことですよ。PC-9801NL(ノートパソコンね。68ノートがあれば最高なんだがなあ)で原稿を書いていたときのことなんです。

このノートにはレジャー機能つてのがあって、電源スイッチを切っても、もう一度電源を入れるとそのまま復活することができるのです。ところがある有名なサードパーティ製のメモリを入れたら、電圧降下が起きてるのか何だか知らないけど、たまにレジャーが失敗するようになったのですよ。怖いなあと思うでしょ？ そのうちとうとうくらってしまった。しかも最悪のパターン。RAMドライブまで初期化されて……(普通はレジャー失敗するだけで、RAMドライブの内容は生き残る)。おかげで原稿の半分ぐらいを失ってしまいました。書き直したけど。メモリーメーカーさん、お願いだから不安定なメモリは出さないで……。しかし、ノート側にもセーフティ機能ぐらい欲しいものですよ。ではまた。

図10 V-I 進行に導音進行が含まれる

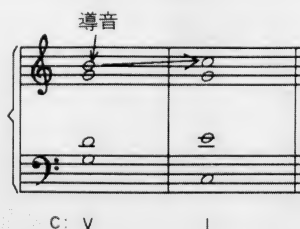
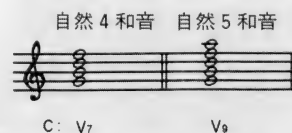


図11 Vの上に自然4和音・自然5和音が構成できる





# (善)のゲームミュージックでバビンチョ



西川善司

## ●9月某日

X68000の調子が悪いのでシャープのサービスセンターに持っていった。受付のおばさんはX68000の取手を持ってぐりりと半回転させ、背面パネルを指でなぞりながら「えーと型番は、と」とつぶやいて、必要事項を「修理預かり証」に書き始めた。しかし、しばらくして私が受け取った「修理預かり証」の型番のところには、

RS-232C

と書いてあった。このおばさんはあまりパソコンの修理の受付には向いていないかもしれないと思った。

## ●9月某日

友人宅でお茶を入れてもらっていた。「あのお菓子買ったんだけど食べる？ ほら、あの、あれ」頭のなかにはその映像が浮かんでいるようだが、名前が思い浮かばないらしい。すっきりしない表情でついにお菓子の容貌を語り出した。「あの、バナナの皮にさ、バナナの実が入ったやつよ」

私はそれはバナナそのものだと思ったが、出てきたお菓子は、バナナの実がカステラの皮に包まれたものだった。

## ●9月某日

ヨドバシカメラにデータディスクマンを修理に出しにいった。カウンターには店員がいて、なにやらあご髭の生えたジョージ・ルーカスみたいな白人男性の客と悪戦苦闘中だった。

白人男性「コノ、STROBE(カメラのストロボ)修理シタイ」

店員「修理するより新しく買ったほうが安いですよ」

ルーカスは片言の日本語は話せるが、店

員の流暢な日本語は理解できないらしい。店員は修理に非常に時間がかかる、ということ伝えてルーカスに諦めさせようとしたらしく、3週間かかるという意味(と思われる)で、

「Three weeks ago」

を繰り返していった。しかし、これでは3週間前という意味になってしまう。一向に間違いに気づかず、手のあいたもう1人の店員までもが舌を丸めて、

「Three weeks ago」

を連発し、店内は「Three weeks ago」合唱団の演奏会になってしまった。

ルーカス「サン週間マエ……ですか?」

ルーカスは混乱する一方だった。

\* \* \*

## ●餓狼伝説SPECIAL/SNK

新世界楽曲雑技団

CD: PCCB-00138

1,500円(税込)

ポニーキャニオン

10/21発売

本家のスーパーより一足先に登場したこのゲーム、なかなかの出来映えで安定した人気を獲得している。しかし、NEO・GEOって格闘ゲームマシンなの? なんていう皮肉が出てくるほど格闘タイプのゲームが多いよね。

さて、登場キャラクターが一気に増えた関係でBGMも増えている。餓狼伝説2から持ち越されたキャラのBGMはほとんどそのまんまの状態だが、餓狼伝説1のキャラクターのBGMは新曲または、より洗練されたアレンジのものになっている。SE&VOICEはもちろんすべてを収録。餓狼伝説2のCDを持っている人は内容がかなりオーバーラップしている印象を受けるかも。

お勧め度 7

## ●WORLD HEROES 2

~IMAGE ALBUM~ /SNK・ADK

ポニーキャニオン

10/21発売

CD: PCCB-00137

2,500円(税込)

「サムライスピリッツ」「餓狼伝説スペシャル」の登場で神隠しにあったかのように姿を消してしまった「ワールドヒーローズ2」だが、イメージアルバムが発売となった。ゲーム中のBGMを全曲アレンジバージョンにて収録。民族音楽系アレンジ、ジュリアナ系アレンジ、TMNモドキ、ボーカルアレンジ……多彩なアレンジで聴き手を飽きさせないような努力がみられる。またどんな録音方法をとったのか知らないが、アンビエンス系のエフェクト効果が素晴らしく、音像、音場が明確で心地よい。

お勧め度 8

## ●ファルコム エンディング・コレクション キングレコード

10/21発売

CD: KICA-1132~3

4,200円(税込)

歴代ファルコムゲーム(イース1からアドバンスド・ロードモナークまで)のエンディング曲を収録した2枚組のアルバム。

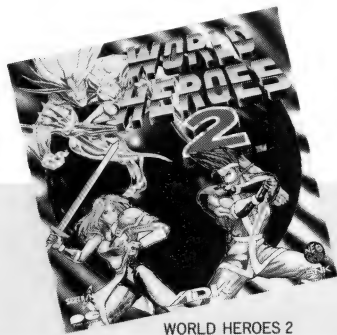
DISC1はオリジナルサウンド、DISC2にはJ.D.K.BANDによるアレンジ演奏が収録されている。PSGとFM音源による演奏のオリジナルサウンドは、それぞれのゲームをクリアしたときの感動を再び思い起こさせてくれる。J.D.K.によるアレンジサウンドは、おとなし目のインストアレンジにとどめられ(!?), 最後まで滞りなく聴き流せる環境音楽のような仕上がりになっている。

お勧め度 7

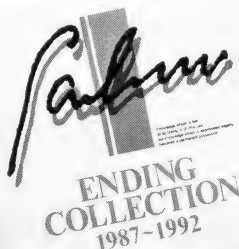
## 終わりに

先月、「おまけ」として紹介したCD「GREAT WALL」の購入方法について、神奈川県の中村圭介君、栃木県の竹原充君ほかからお便りをいただいた。どうもありがとう。購入希望の方は下記の住所に問い合わせをしてほしい。また、この件に関してはOh!X編集部は一切関係していない。こちらに問い合わせをしても何もわからないので、そこそこよろしく。

〒195 東京都町田市三輪14-17相原様方  
TROUBADOUR RECORD事業所  
それではまた来月。



WORLD HEROES 2

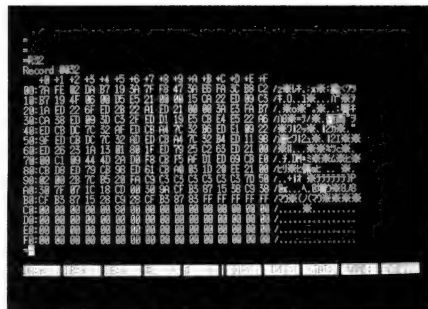


ファルコム エンディング・コレクション



# THE SENTINEL

〈対応機種一覧〉 ●MZ-80 K/C/700/1500 ●MZ-80 B/  
2000 ●MZ-2500/286i ●X1 ●X1 turbo/Z ●PC-8001/  
8801/88 ●SMC-777/C ●PASOPIA/5 ●PASOPIA/7 ●  
FM-7/77/AV ●MSX/2/2+/turbo R ●PC-286/386/486/  
9801/98/982i ●X 68000/X 68030  
掲載されたプログラムの利用には各機種用のS-OS  
“SWORD”システムが必要です。



がるでしょう。しかしそれでは面白くありませんからね。

速度的に厳しければ、別にリアルタイムにこだわる必要もないはず。プレイヤーが次にどんな行動を起こすべきか考えさせ、そして、イベントによってプレイヤーのゲームに対する思い入れを増すことができる。いいのですから。

そのためには、ストーリーとかある特定のアイテムを必ず得なければならない、などの制限をもつものはいけません。あくまでも、目標はそれぞれプレイヤーが探して、それぞれに楽しめるものが望ましいですね。いきなりゲームの世界に放り出されて、手探りで解読しながら遊ぶのもなかなか楽しいものです。自分の手で遊んでいるという実感がもてるゲーム、いまだにからの非常に古いタイプのゲームです。しかし、そのゲーム性は現在のゲームにも受け継がれているでしょう。

とまあ、言葉で表現するのは非常に簡単ですが、最初からあきらめず頭をひねってみましょう。面白いアイデアがあれば、どしどしこのTHE SENTINELにお送りください。お待ちしております。

## 第137部 S-OSで学ぶZ80マシン語講座(1)

### ●S-OSとアセンブラ

今月から「S-OSで学ぶZ80マシン語講座」が開始されました。

ディスクダンプエディタを題材に、S-OSの使い方、Z80アセンブラの使い方をレクチャーしていこうというものです。

いまだにこのようなZ80マシン語講座をやろうとしているのは、新しく仲間に加わったMSX用S-OS“SWORD”の発表による反響を反映しなければならない、と判断したためです。システムを使うためには、それなりのルールがあります。MSXでも使える！とはいっても、ほとんどの人が初めてS-OSを使うはず。右も左もわからない状態のまま、ただシステムだけがそこにある、というような状況を作り出したいのです。

もちろん、現在8ビット機ユーザーで、一度挫折してしまった人もこの機会に再挑戦してもらいたいものです。確かにアセンブラには、とっつきにくい部分もありますが、自分のものにしてしまえばマシンを自由自在に操ることも可能です。

特にS-OSは、究極のマシン語モニタの言葉どおり、必要最低限のコマンド、システムコールしか用意されていません。しかも、基本的にそれらは、アセンブラレベルで使用することを前提としています。

SLANGなどの高級言語を使うのもいいですが、マシンの隅々まで手が届くアセンブ

ラの世界を覗いて見るのも損はないはずです。ぜひ、挑戦してみてください。

### ●これからのTHE SENTINEL (4)

さてさて、10月号のTHE SENTINELでかなり軽いノリで読者意見の募集を行っていましたが、まだ発表できるほど集まっていません。

THE SENTINEL WORLDとしての新装開店、もう少し先のことになりそうです。もしかしたらそのまま自然消滅……なんてことはしたくありませんから、ぜひぜひ読者の皆さんの声をお聞かせください。

そんななかで、第1回のテーマにいち早く飛びついたのが、東京都の相沢栄樹さん。すでにオリジナルシステムの開発を始めています。せっかくやる気を出したのに、出し抜かれてはもうだめだね、と考えている人。あきらめずにアイデアだけでもお送りください。

そして、第2回のテーマは「ROGUEタイプの自動迷路生成、何度でも手軽に遊べるRPG」です。以前から「これこそS-OS向きの題材だと思うんだけどなあ」と考えていたのですが、すでに誰か制作していないのでしょうか。単純ななかにも光る戦略性、飽きのこないゲーム性などが必要になるはずですから、テーマとしてはかなり厳しいといえます。ただ単に乱数で迷路の生成を行って、同じく乱数で各種イベントを設定するだけで、一応それらしいものが出来上

## 1993■インデックス

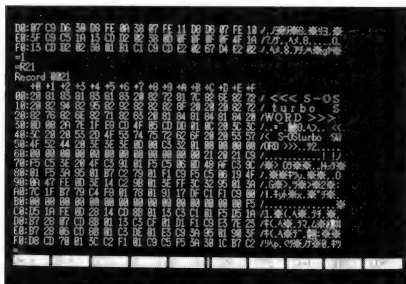
- 93年1月号――
- 第128部 EDC-Tの拡張
- 93年2月号――
- 第129部 BLACK JACK
- 93年3月号――
- 第130部 シューティングゲームコアシステム作成法 (1)
- 93年4月号――
- 第131部 シューティングゲームコアシステム作成法 (2)
- 93年5月号――
- 第132部 シューティングゲームコアシステム作成法 (3)
- 93年6月号――
- 第133部 REVERSI
- 93年7月号――
- 特別付録 MSX用S-OS“SWORD”
- 93年8月号――
- 第134部 MACINTOSH-C再掲載
- 93年9月号――
- 第135部 7並べ
- 特別付録 SLANG再々掲載
- 93年10月号――
- 第136部 シューティングゲームコアシステム作成法 (4)

全機種共通  
S-OS“SWORD”要

# S-OSで学ぶ Z80マシン語 講座(1)

Itou Masahiko  
伊藤 雅彦

いろいろな言語が発表されてきたS-OSですが、そのポテンシャルを最大限に引き出せるのはやはりアセンブラです。今月から始まるマシン語講座をもとに、ぜひマスターしましょう。



このTHE SENTINELのコーナーは、もはや覆い隠せないほどにパワーダウンしています。

Z80をCPUに持つパソコンならば、どの機種でも同じプログラムを動かせるようにしようと生まれたS-OSですが、Z80マシンが使われなくなっていくなかでS-OSが活用される場もなくなってきています。みんな16ビットや32ビットのマシンを持つてのに、あえて8ビット機を前提としたS-OSを使おうと思う人はそうはいませんよね。それはとても自然なことです。

でも、そんな自然の流れに素直に従うのはちょっと面白くありません。ここでZ80マシンをもっと使おうって呼びかけてみようと思います。なぜって、8ビット機だっていまでも十分面白いおもちゃになるじゃないですか。

プログラミングユーザーにとって、パソコンっていうのはおもちゃなんです。積み木みたいな。もういじっているだけで楽しいんだと。いじっているうちに、あんなことやってみよう、こんなことやってみようっていろいろ浮かんできて、プログラムを作っちゃう。小さな子供が積み木を思いっくまに積み上げて、家や門なんかを作って遊んでるような感じでね。

そういう楽しさっていうのは、8ビット機でだって味わえるんです。仕事に使おうとするとちょっと苦しいし、ゲームマシンとしても低級機だけど、プログラミングが楽しめるおもちゃとしては、8ビットでもいまどきのマシンとタメを張れます。特にS-OSっていうのはシンプルだから、システムを理解して遊ぶための決まりごとを心得るのに、そんなに労力はいりません。すぐにでも「こいつで遊んでやろう」っていう気力が湧いてきます。キャラグラだって多彩な表現が可能なのですから。

だから、プログラミングって面白そうだなと思っているあなた。押し入れから8ビット機を引っ張り出してください。友達の家で眠っている8ビット機をかつばってください。中古ショップに転がっている8ビット機を買い叩いてください。そしてS-OS“SWORD”を立ち上げたら、プログラミングの楽園があなたを待っています。財布を痛めずに望みが叶えられるなんて、この幸せ者っ！

## マシン語です

今月から短期集中連載ということで、Z80マシン語入門講座をやります。やっぱりパソコンやるならマシン語の心得があったほうが、パソコンに対する理解が違ってくるんです。それにマシン語を使えば、自分のパソコンのパワーを100%発揮させるもさせないも、すべて自分次第。これがホビープログラマにはたまらない魅力なんです。

この講座では実際に1本のプログラムを作っていきます。その様子を伝えながらプログラミングの進め方を感じ取ってもらおうというわけです。よって、個々の命令の説明は最低限必要な程度にとどめますので、この講座を読んでマシン語をやろうと決意した方は、自分で命令解説本を買って基礎知識を広げてください。

では、今月はイントロダクションということで、Z80にできることはなんなのかってところをお話ししましょう。Z80は、

1) ある場所にあるデータをほかの場所に写し取る

ここで「場所」とは、メモリやレジスタのことです。レジスタというのは、Z80CPUの中にあるデータの記憶場所です。なぜこんなものがあるのかは、おいおいわかることです。続いて、

2) ある場所にあるデータを、足し算や引き算や論理演算その他いろいろなことをして、加工する

3) データ間の大小比較などをし、その結果によって違うプログラムを実行させる

さらに補足するなら、

補) データを扱うときの単位は8ビットか16ビット

こんなもんでしょうかね。要するに、Z80はメモリなどのデータをいじくり回すことができるってことです。そして、どういう具合にいじくり回すかを決めるのが、プログラムというわけなんです。

メモリをいじくるばかりだったら、画面に文字を表示したり、キー入力したりするのはどうやってやるのってことになるわけですが、それも基本的にはメモリの読み書きと同じです。特定のアドレスを読み書きすると、そこがCRTやキーボードなどの



装置とつながっていて、データのやりとりができるんです。Z80の場合だと、メモリを読み書きするためのアドレスと、装置とデータのやりとりをするためのアドレスは、CPU レベルではっきり区別されていて、読み書きするための命令も違うんですけどね。

具体的にどういうやりとりをすれば画面に文字を表示できるのかというのは、それぞれのパソコンによって違います。ですから、知りたければ各機種ハードの解説書を手に入れるしかありません。機種によって、同じことをするのに違うプログラムにしくなくちゃならないんですね。MZ-2500用のプログラムがX1で動かないのも、このせいでいいわけです。

でも、しかし、にもかかわらず、S-OSなら全機種で共通に動くプログラムが作れてしまいます。それは「システムコール」と呼ばれる、S-OSシステムの中にあるプログラム集のおかげです。1文字表示するプログラムやなんかサブルーチンの形でたくさん用意してあるんです。1文字表示したかったら1文字表示サブルーチンをコールするようにプログラムしておけば、MZでもMSXでも、各機種用のS-OSの中の各機種用の1文字表示サブルーチンが実行されて、どの機種でも同じように1文字表示ができる仕組みです。システムコールっていうのは、このように機種間の違いを吸収してくれる効能もあるし、さらにこまごまとしたうとうしい処理を引き受けてくれるというありがたい面もあります。うまく活用していきたいものです。各コールの具体的な利用方法は来月以降に説明します。

Z80ができることは、データをいじくり回すこと。とりあえずこのことを理解するのが第一歩です。でも、どうデータをいじくればゲームなどができるのか、入門者にはなかなか想像がつかないと思います。以前にマシン語入門書を読んでみて、データをあれこれいじっているらしいというのはわかったけど、どうやってプログラムを作ったのかどうも掴めなかったという人、これからの講座にご期待ください。

## ■■■■■■■■■■ ディスクとは ■■■■■■■■■■

この講座は実際にプログラムを作りながら進めていくわけですが、どんなプログラ

ムを作るかここでいっておきましょう。ゲームじゃありません。フロッピーディスクのデータを覗いたり書き換えたりするツール、ディスクエディタです。名前は“ADDIE (アディ)”, “ADvanced DIsk Editor” とこじつけました。

S-OSはテープベースでも使えるんですが、このプログラムはディスクユーザーのみが対象となってしまいます (テープ版でもRAMディスクには使えますが)。テープ、QDユーザーの方、申し訳ありません (でもいったい何人いるんだろう)。

以前に発表されたディスクエディタといえば、1986年10月号のDREAMが最も新しいということになりますから、ずいぶん古い話になるわけです。私なんか当時は読者でもなかったんですから (テクノポリスの読者だった、あはは……)。

そこで今回、DREAMの仕様なども参考にしながら、ちゃんと使いものになるディスクエディタを作ります。マシン語講座のネタだからといって、初心者向けのわかりやすいプログラムを組もうという気はさらさらありません。ま、それでもそんなに難

解なものにはならないでしょうし、リストに注釈を多めに入れるつもりではありますから、大丈夫でしょう。

ADDIEの仕様は一応のところ固まっています。コマンド一覧を表1に示します。データがおかしくなったディスクを調べたり、間違って消したファイルを復活させたり、ファイルを整理したりといった作業に役立つようにと、あれこれ考えたものです。あとで仕様を変更することもあります、そのときは勘弁してください。いろいろ複雑な事情があって、どうしてもってときがあるんです。作るのが面倒臭いとか、煩わしいとか、怠けたいとか。

ま、とにかくですね、この講座ではディスクエディタを作りながらマシン語のお勉強をするわけです。となると、フロッピーディスクとはどんなものかということを用意知識として知っておいていただきたいわけです。ディスクを扱うプログラムを解説するってときに、ディスクが何者かわからないっていうんじゃない話できませんから。

フロッピーディスクというのは、あのドクター中松、中松義郎氏が発明したものな

表1 コマンド一覧

(パラメータの数値は任意桁の16進法、[ ] 内のパラメータは省略可)

V [＜デバイス名＞] デバイスの変更。デバイス名省略時は現在のデバイスを表示。	O [＜先頭ファイル番号＞] [[＜最終ファイル番号＞] ＜移動先ファイル番号＞] ディレクトリのファイル登録順の並べ替え。先頭ファイル番号から最終ファイル番号までのファイルを移動先ファイル番号へ移動する。最終ファイル番号省略時は先頭ファイル番号のファイルのみ移動。移動先ファイル番号省略時は先頭へ移動。全パラメータ省略時はガーベジコレクションを行う。
R [＜レコード番号＞] セクタのダンプ表示。	K <ファイル番号……> ファイルの削除。
W [＜レコード番号＞] セクタのデータ書き換え。	S <先頭レコード番号> <最終レコード番号> <検索データ列> データ列の検索。
F クラスタの使用状況の表示。80桁モードのときはFATの生データも表示。	T [＜フィルター＞] ディスクの最適化。全ファイルコピーを行った際のコピー先ディスクのように、クラスタの使用状態を整える。フィルター指定時は未使用クラスタをフィルター (1バイトデータ) で埋める。
D [＜ファイル番号＞] [＜変更項目指定> ＜変更データ>] ディレクトリデータの表示。ファイル番号省略時は全ファイルのディレクトリデータを表示。変更項目指定時はディレクトリデータを書き換える。変更項目指定は、A (ファイル属性)、S (ファイル長)、T (先頭アドレス)、E (実行アドレス)、N (ファイル名) の5種類。 DIT800 でファイル番号1のファイルの先頭アドレスが8000 <sub>H</sub> になる。	P プリンタ出力のON/OFFの切り替え。
C [＜ファイル番号＞] [＜クラスタ番号……>] クラスタ連鎖の表示。ファイル番号省略時は全ファイルのクラスタ連鎖を表示。クラスタ番号指定時はクラスタ連鎖を変更。	H ヘルプメニューの表示。
	Q 終了。

んだそうで。テレビで見ると変わり者のおやじのように見えるんですが、こんな実用的で堅実な発明もしてたんですね。だとすると、ジャンピングシューズも結構なものかもしれないかもしれません。

そんな生まれのフロッピーディスク、磁性体を塗った円盤にデータを同心円状に記録しているというのは、皆さんご存じのとおりです。アナログレコードやCDは螺旋状に記録されていますが、フロッピーディスクは円が何重にも重なっています。減点パパだったら大喜びするところですよ。この円をトラックと呼び、2Dディスクだと普通40重円になっているので40トラック。でもそれが裏表ありますから、全部で80トラックです。

ひとつのトラックはいくつかに区切られていて、そのひと区切りをセクタと呼びます。2Dでは普通1トラックを16セクタに区切っています。このとき、1セクタの中に256バイトのデータを記録できますから、ディスク全体の記憶容量は、 $256B \times 16 \text{セクタ} \times 80 \text{トラック} = 320K \text{バイト}$ となります。いまだき2Dを例に出すと、なんだか多少恥ずかしい気もしてきますが、S-OSは2Dを想定したディスク管理方式になっていますからね。

それで、このセクタというのは結構重要な単位です。フロッピーディスクの読み書きは、セクタ単位で行われるのです。1バ

イトだけ読みたいと思っても、1セクタ分256バイトを読まなくてはいけないし、書き込みも同じです。フロッピーディスクはランダムアクセス可能なデバイスといわれますが、1セクタ内のデータに関しては、任意のデータだけをいきなりアクセスすることはできないわけです。

でも、セクタ単位でなら問題なく、ランダムにアクセスできます。あっちのセクタを読んで、今度はこっちのセクタを読み、そっちのセクタに書いて、という具合に自由自在です。このとき、あっちのセクタとかいってると、あっちってどっちなんだということになりますから、すべてのセクタに通し番号がついています。これをレコード番号といいます。

さて、ここまではフロッピーディスクそのものの話でした。S-OSではこのような記録メディアにファイルを記録しているわけです。ではそのファイルはどのように記録されているのでしょうか。

S-OSのディスク管理方式は、X1の2Dディスクの場合の方式とほぼ同じです。重要なのがディレクトリとFAT (File Allocation Table) というやつ。ディレクトリはS-OSのモニタのDコマンドで表示されるようなファイル情報が記録されているところです。ひとつのファイルにつき32バイトの領域が使われ、図1のような各種情報が書き込まれています。+18バイト目からファ

イル長・先頭アドレス・実行アドレスと並んでいます。これはDコマンドでの表示のされ方と少し違っていますから注意してください (Dコマンドでの表示は先頭アドレス・最終アドレス・実行アドレス)。

また、+30バイト目に先頭クラスタ番号とありますが、クラスタというのは16セクタをひとまとめたものです。レコード番号0~15が第0クラスタで、以下16セクタごとにクラスタ番号がつけられています。Dコマンドでも“\$xx ClustersFree”って表示されますよね。2Dディスクでは1クラスタは1トラックと同じことになります。

ディスクの使用状態はクラスタ単位で管理されます。このクラスタはこのファイルの内容を記録しているとか、未使用だとか、そういった情報が書かれているところがFATです (表2)。FAT領域の大きさは128バイトで、先頭バイトから第0クラスタ、第1クラスタ……、第127クラスタの情報が書き込まれています。その情報の意味は表2のとおりです。簡単にいうと、クラスタが使用されているかどうか、使用されているか、データの続きがどのクラスタに入っているか、ということが書いてあるわけです。

さて、ここでわかるのは、この管理方式では128クラスタしか管理できないということです。2Dディスクなら80クラスタですから問題ないんですが、2DDでは160クラスタで、32クラスタ分足りません。2HDではなお足りなくなります。というわけで、S-OSでは2DDや2HDが使えるX1turbo版でも、メディアの容量分フルに記録することはできません。128クラスタ=512Kバイトまでです。ま、いいじゃないですか。

FATはディスクの第14レコードに記録されています。ディレクトリのはうは第16レコードから、メディアの容量により最大第31レコードまでのセクタが使われています。この記録位置は変更することもできますが、そのようなことをする必要はまずな

図1 ディレクトリ

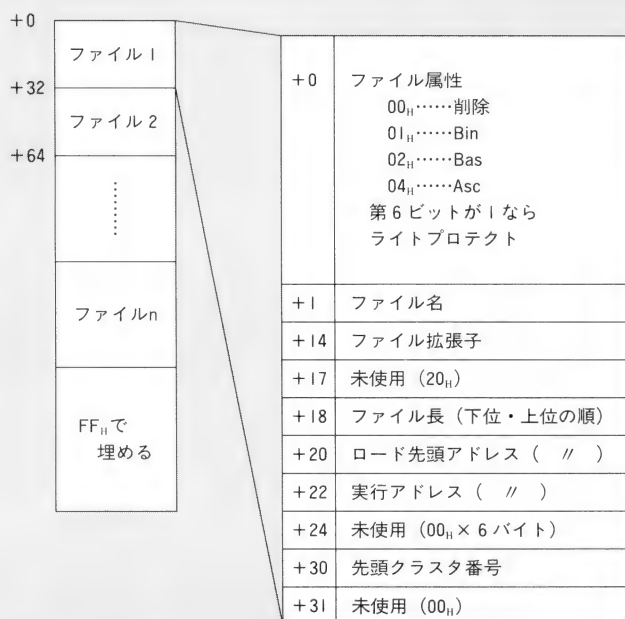


表2 FAT情報

00 <sub>H</sub>	未使用クラスタ。
01 <sub>H</sub> ~7F <sub>H</sub>	使用中。データの続きがあり、それが記録されているクラスタのクラスタ番号を表す。
80 <sub>H</sub> ~8F <sub>H</sub>	使用中。データの続きはなく、7F <sub>H</sub> を引いた値がこのクラスタ内で使用されているセクタ数を表す。



いといっでいでしょう。S-OSで使うディスクには、これらのセクタにディレクトリとFATが存在してはいけません。おわかりいただけたでしょうか。

## ■■■■■■■■ サンプル版をどうぞ ■■■■■■■■

さて、今月は最後にADDIEのサンプル版をお届けしましょう。サンプルですから一部の機能しか使用できません。使用できないというより、まだ作っていないんですが。使えるのは、V、R、Qコマンド。セクタのダンプ表示しかできないってことです。

こんなセコいサンプル版を掲載する理由は2つあります。1つは、ディスクの中を実際にダンプして覗いてみれば、ディスクの管理方法が理解しやすいだろうということ。ディレクトリやFATなどをダンプして、納得してもらえればと思います。

それから2つめの理由は、プログラムリストの入力、アセンブル手順を覚えてもらうためです。リスト1を見てください。これはADDIE サンプル版のソースリストですが、マシン語でプログラムを作るときは、こういったアセンブリソースプログラムをエディタで入力して、アセンブラでそのソースプログラムを実行プログラムに変換することになります。この作業を自分でひと通り体験してみてください。

ここで必要なアプリケーションとして、エディタとアセンブラを用意してください。

この2つがないとマシン語プログラムの開発ができませんからね。どちらもいまだに何種類か発表されています。エディタではE-MATE、WINER、TED-750、EDC-Tの面々。どれを使ってもかまいません。アセンブラのほうでは、標準アセンブラといえるREDAを始め、ZEDA、OHM-Z80、WZDがあります。WZDはWLKとペアで使わないと実行プログラムが作られません。

また、REDAとZEDAはエディタを内蔵していますから、これひとつでひと通りのマシン語開発ができます。

この連載では、REDAおよびZEDAでアセンブルできるソースを掲載します。でもたぶんOHM-Z80でもアセンブルできると思います。WZDではちょっと変更を加える

必要があるんですが、この連載ではWZDを使うことは想定しないことにします。

エディタとアセンブラを用意したら、リスト1をエディタで入力してください。漢字の注釈はカタカナにしたり、省略したりしてかまいません。入力し終わってセーブしたら、今度はアセンブラを使って実行プログラムを作ってください。B000<sub>H</sub>番地から、3000<sub>H</sub>番地用のプログラムが生成されるはず。それをいったんセーブして、3000<sub>H</sub>番地にロードし直してください。3000<sub>H</sub>番地をコールすれば実行開始です。どうでしょう、ちゃんとできましたか？

次回からマシン語講座の本編に入ります。どうなることやら私にもわかりませんが、とりあえず楽しみに。

## 連載のこれから

なにも考えていません。なんていったら真面目に期待してくれた人に怒られそうだから、こちらも真面目にお話ししましょう。

まず、記事中にもあるとおり基本方針は、S-OSの使い方を学び、そして、Z80のマシン語も理解してしまおうというものです。

とはいっても、連載開始から結構大きなソースリストを出すところから推測できるように、Z80の個々の命令を1つひとつ詳細に解説していくつもりはありません。なぜなら、命令の詳細を知りたいければ、参考書籍を1冊買ってくればすむことです。

重要なのは、個々の命令ではなく作ろうと思っているプログラムを、どのようにして自分の知っているコードに落とすか、であると思うん

です。もちろん、そのときには命令をすべて把握しているのが望ましいのですが、知らないなら知らないなりになんとかするものです。

私だって最初のうちは、まごまごしていたものです。そして、徐々に使える命令を増やしていき、現在では、多少プログラムができる人間、と自分で思えるくらいになりました。

また、こういった連載でそれぞれの命令を何回かに分けて解説しても、あとあとになって必要になったからといって、わざわざ雑誌をひっくり返して目的の命令を捜し出すのも面倒です。参考書籍を見たほうが断然早いでしょう。

そういうことで、いずれ役に立つと信じて、今月号のサンプルプログラムを、がんばって入力してしましましょう。

## リスト1

```
0000 1 ;
0000 2 ; A D D I E (ADvanced Disk Editor)
0000 3 ;
0000 4 ; Program : Masahiko Ito
0000 5 ;
0000 6 ; Oh!X 11/'93 Version
0000 7 ; (Command:V,R,Q)
0000 8 ;
0000 9 #PRINT: EQU 01FF4H
0000 10 #PRNTS: EQU 01FF1H
0000 11 #LTNL: EQU 01FEEH
0000 12 #MSX: EQU 01FE5H
0000 13 #MPRNT: EQU 01FE2H
0000 14 #LPTON: EQU 01FD9H
0000 15 #LPTOF: EQU 01FD6H
0000 16 #GETL: EQU 01FD3H
0000 17 #PAUSE: EQU 01FC7H
0000 18 #BELL: EQU 01FC4H
0000 19 #PRTHX: EQU 01FC1H
0000 20 #PRTHL: EQU 01FBEH
0000 21 #ASC: EQU 01FBBH
0000 22 #HEX: EQU 01FB8H
0000 23 #DRDSB: EQU 02000H
0000 24 #FLGET: EQU 02021H
0000 25 #RDVSW: EQU 02024H
0000 26 #ERROR: EQU 02033H
0000 27 ;
0000 28 #KBFA: EQU 01F76H
0000 29 #DTBUF: EQU 01F64H
0000 30 #DSK: EQU 01F5DH
0000 31 #WIDTH: EQU 01F5CH
0000 32 ;
0000 33 PROMPT: EQU ' '
```

```
0000 34 ;
0000 35 ;
0000 36 OFFSET 0B000H-03000H
0000 37 ORG 03000H
0000 38 ;
0000 39 ; Main
0000 40 ;
0000 41 MAIN:
0000 42 LD A, (#WIDTH)
0000 43 SUB 40
0000 44 JR Z, MAIN1
0000 45 LD A, 1
0000 46 MAIN1:
0000 47 LD (WIDMODE), A
0000 48 ;
0000 49 CALL #RDVSW
0000 50 CP 'A'
0000 51 JR C, MAIN2
0000 52 CP 'E'+1
0000 53 JR C, MAIN3
0000 54 MAIN2:
0000 55 LD A, 'A'
0000 56 MAIN3:
0000 57 LD (DEVICE), A
0000 58 ;
0000 59 XOR A ; A=0
0000 60 LD (PTRSW), A
0000 61 LD (WRCHK), A
0000 62 LD L, A
0000 63 LD H, A
0000 64 LD (RECORD), HL
0000 65 ;
0000 66 MAIN4:
```

```

3028 CD D6 1F 67 CALL #LPTOF
302B 3E 3D 68 LD A,PROMPT
302D CD F4 1F 69 CALL #PRINT
3030 ED 5B 76 70 LD DE, (#KBFD)
3033 1F
3034 CD D3 1F 71 CALL #GETL
3037 1A 72 LD A,(DE)
3038 FE 3D 73 CP PROMPT
303A 20 EC 74 JR NZ,MAIN4
303C
303C 13 75 ;
303D ED 53 D9 76 INC DE
3040 31 77 LD (KBPTR),DE
3041 CD B6 31 78 CALL SPCUT
3044 CD C3 31 79 CALL CAPITAL
3047 21 28 30 80 LD HL,MAIN4
304A E5 81 PUSH HL ; RET で MAIN4 にジャンプさ
せる細工
304B FE 56 82 CP 'V'
304D CA 8C 30 83 JP Z,VCOM
3050 FE 52 84 CP 'R'
3052 CA BD 30 85 JP Z,RCOM
3055 FE 57 86 CP 'W'
3057 CA 3E 31 87 JP Z,WCOM
305A FE 46 88 CP 'F'
305C CA 3F 31 89 JP Z,FCOM
305F FE 44 90 CP 'D'
3061 CA 40 31 91 JP Z,DCOM
3064 FE 43 92 CP 'C'
3066 CA 41 31 93 JP Z,CCOM
3069 FE 4F 94 CP 'O'
306B CA 42 31 95 JP Z,OCOM
306E FE 4B 96 CP 'K'
3070 CA 43 31 97 JP Z,KCOM
3073 FE 53 98 CP 'S'
3075 CA 44 31 99 JP Z,SCOM
3078 FE 54 100 CP 'T'
307A CA 45 31 101 JP Z,TCOM
307D FE 48 102 CP 'H'
307F CA 46 31 103 JP Z,HCOM
3082 FE 50 104 CP 'P'
3084 CA 47 31 105 JP Z,PCOM
3087 FE 51 106 CP 'Q'
3089 C0 107 RET NZ
308A
308A E1 108 ;
308B C9 109 POP HL ; 細工を解除
308C 110 RET
308C 111 ;
308C 112 ; V Command
308C 113 ;
308C 114 VCOM:
308C CD B6 31 115 CALL SPCUT
308F CD C3 31 116 CALL CAPITAL
3092 FE 41 117 CP 'A'
3094 38 10 118 JR C,VCOM1
3096 FE 46 119 CP 'E'+1
3098 30 0C 120 JR NC,VCOM1
309A 121 ;
309A 32 D5 31 122 LD (DEVICE),A
309D AF 123 XOR A ; A=0
309E 32 D8 31 124 LD (WRCKB),A
30A1 6F 125 LD L,A
30A2 67 126 LD H,A
30A3 22 D6 31 127 LD (RECORD),HL
30A6 128 ;
30A6 129 VCOM1:
30A6 CD CC 31 130 CALL PRTRSET
30A9 CD E2 1F 131 CALL #MPRNT
30AC 44 65 76 132 DM 'Device '
30AF 69 63 65
30B2 20
30B3 00 133 DB 0
30B4 3A D5 31 134 LD A,(DEVICE)
30B7 CD F4 1F 135 CALL #PRINT
30BA C3 EE 1F 136 JP #LTNL ; = CALL #LTNL : RET
30BD 137 ;
30BD 138 ; R Command
30BD 139 ;
30BD 140 RCOM:
30BD CD 8B 31 141 CALL PARAMETER
30C0 DA C4 1F 142 JP C,#BELL ; = IF C (CALL #BELL : RET)
30C3 143 ;
30C3 20 03 144 JR NZ,RCOM1
30C5 2A D6 31 145 LD HL,(RECORD)
30C8 146 ;
30C8 147 RCOM1:
30C8 CD CC 31 148 CALL PRTRSET
30CB CD E2 1F 149 CALL #MPRNT
30CE 52 65 63 150 DM 'Record '
30D1 6F 72 64
30D4 20
30D5 00 151 DB 0
30D6 CD BE 1F 152 CALL #PRTHL ; レコード番号表示
30D9 CD EE 1F 153 CALL #LTNL
30DC 154 ;
30DC 3A D5 31 155 LD A,(DEVICE)
30DF 32 5D 1F 156 LD (#DSK),A
30E2 EB 157 EX DE,HL
30E3 2A 64 1F 158 LD HL, (#DTBUF)
30E6 3E 01 159 LD A,1
30E8 CD 00 20 160 CALL #DRDSB ; セクタ読み込み
30EB DA 33 20 161 JP C,#ERROR ; = IF C (CALL #ERROR : RET)
30EE 162 ;
30EE 13 163 INC DE
30EF ED 53 D6 164 LD (RECORD),DE
30F2 31

```

```

30F3 3E 01 165 LD A,1
30F5 32 D8 31 166 LD (WRCKB),A
30F8 167 ;
30F8 CD F1 1F 168 CALL #PRNTS
30FB CD F1 1F 169 CALL #PRNTS
30FE 170 ;
30FE 26 00 171 LD H,0
3100 05 10 172 LD B,16
3102 3A D4 31 173 LD A,(WIDMODE)
3105 B7 174 OR A
3106 20 02 175 JR NZ,RCOM2
3108 05 08 176 LD B,8
310A 177 RCOM2:
310A CD E2 1F 178 CALL #MPRNT
310D 20 2B 179 DM ' +'
310F 00 180 DB 0
3110 7C 181 LD A,H
3111 CD BB 1F 182 CALL #ASC
3114 CD F4 1F 183 CALL #PRINT ; スケール表示
3117 24 184 INC H
3118 10 F0 185 DJNZ RCOM2
311A 186 ;
311A CD EE 1F 187 CALL #LTNL
311D 188 ;
311D AF 189 XOR A ; A=0
311E 32 DC 31 190 LD (DUMPOS),A
3121 2A 64 1F 191 LD HL, (#DTBUF)
3124 192 RCOM3:
3124 0E 10 193 LD C,16
3126 194 RCOM4:
3126 CD C7 1F 195 CALL #PAUSE
3129 3D 31 196 DW RCOM5
312B 3E 2F 197 LD A,'/'
312D CD 48 31 198 CALL DUMP ; セクタデータ1行表示
3130 0D 199 DEC C
3131 20 F3 200 JR NZ,RCOM4
3133 201 ;
3133 3A DC 31 202 LD A,(DUMPOS)
3136 B7 203 OR A
3137 C8 204 RET Z
3138 205 ;
3138 CD 21 20 206 CALL #FLGET
313B 18 E7 207 JR RCOM3
313D 208 ;
313D 209 RCOM5:
313D C9 210 RET
313E 211 ;
313E 212 ; W Command
313E 213 ;
313E 214 WCOM:
313E C9 215 RET
313F 216 ;
313F 217 ; F Command
313F 218 ;
313F 219 FCOM:
313F C9 220 RET
3140 221 ;
3140 222 ; D Command
3140 223 ;
3140 224 DCOM:
3140 C9 225 RET
3141 226 ;
3141 227 ; C Command
3141 228 ;
3141 229 CCOM:
3141 C9 230 RET
3142 231 ;
3142 232 ; O Command
3142 233 ;
3142 234 OCOM:
3142 C9 235 RET
3143 236 ;
3143 237 ; K Command
3143 238 ;
3143 239 KCOM:
3143 C9 240 RET
3144 241 ;
3144 242 ; S Command
3144 243 ;
3144 244 SCOM:
3144 C9 245 RET
3145 246 ;
3145 247 ; T Command
3145 248 ;
3145 249 TCOM:
3145 C9 250 RET
3146 251 ;
3146 252 ; H Command
3146 253 ;
3146 254 HCOM:
3146 C9 255 RET
3147 256 ;
3147 257 ; P Command
3147 258 ;
3147 259 PCOM:
3147 C9 260 RET
3148 261 ;
3148 262 ; DUMP
3148 263 ;
3148 264 ; in ---- A = キャラクタ表示する('/') / し
ない(0)
3148 265 ; HL = 表示開始アドレス
3148 266 ; out --- HL = 表示最終アドレス + 1
3148 267 ; break - F, A, B, DE
3148 268 ;
3148 269 DUMP:

```



```

3148 32 DD 31 270 LD (DUMPWK),A
314B 271 ;
314B 3A DC 31 272 LD A,(DUMPOS)
314E F5 273 PUSH AF
314F CD C1 1F 274 CALL #PRTHX
3152 3E 3A 275 LD A,';'
3154 CD F4 1F 276 CALL #PRINT
3157 277 ;
3157 3A D4 31 278 LD A,(WIDMODE)
315A B7 279 OR A
315B 06 08 280 LD B,8
315D 28 02 281 JR Z,DUMP1
315F 06 10 282 LD B,16 ; B = 表示バイト数
3161 283 ;
3161 284 DUMP1:
3161 F1 285 POP AF
3162 80 286 ADD A,B
3163 32 DC 31 287 LD (DUMPOS),A
3166 288 ;
3166 11 DE 31 289 LD DE,DUMPWK+1
3169 290 DUMP2:
3169 7E 291 LD A,(HL)
316A 23 292 INC HL
316B F5 293 PUSH AF
316C CD C1 1F 294 CALL #PRTHX
316F CD F1 1F 295 CALL #PRNTS
3172 F1 296 POP AF
3173 297 ;
3173 FE 20 298 CP 020H
3175 30 02 299 JR NC,DUMP3
3177 3E 2E 300 LD A,'.'
3179 301 DUMP3:
3179 12 302 LD (DE),A
317A 303 ;
317A 13 304 INC DE
317B 10 EC 305 DJNZ DUMP2
317D 306 ;
317D AF 307 XOR A ; A=0
317E 12 308 LD (DE),A
317F 11 DD 31 309 LD DE,DUMPWK
3182 CD E5 1F 310 CALL #MSX
3185 CD F1 1F 311 CALL #PRNTS ; 全角文字に対応
3188 C3 EE 1F 312 JP #LTNL ; = CALL #LTNL: RET
318B 313 ;
318B 314 ; PARAMETER
318B 315 ;
318B 316 ; out --- Z = パラメータがない(1)/ある(0)
318B 317 ; Cy = パラメータが16進数として無効
(1)/有効(0)
318B ; HL = パラメータの値 (Z=0 かつ Cy=
0 の時)
318B 318 ;
318B 319 ; break - F, A, DE, HL
318B 320 ;
318B 321 PARAMETER:
318B CD B6 31 322 CALL SPCUT
318E B7 323 OR A
318F C8 324 RET Z ; パラメータがない
3190 325 ;
3190 21 00 00 326 LD HL,0
3193 ED 5B D9 327 LD DE,(KBPTR)
3196 31
3197 328 PARAMETER1:
3197 CD C3 31 329 CALL CAPITAL
319A CD B8 1F 330 CALL #HEX
319D 38 13 331 JR C,PARAMETER3 ; パラメータが16進数とし
て無効
319F 29 332 ADD HL,HL
31A0 29 333 ADD HL,HL
31A1 29 334 ADD HL,HL
31A2 29 335 ADD HL,HL ; 16倍
31A3 85 336 ADD A,L
31A4 6F 337 LD L,A
31A5 1A 338 LD A,(DE)
31A6 B7 339 OR A
31A7 28 05 340 JR Z,PARAMETER2
31A9 13 341 INC DE
31AA FE 20 342 CP ' '
31AC 20 E9 343 JR NZ,PARAMETER1

```

```

31AE 344 ;
31AE 345 PARAMETER2:
31AE ED 53 D9 346 LD (KBPTR),DE
31B1 31
31B2 347 ;
31B2 348 PARAMETER3:
31B2 3E 00 349 LD A,0
31B4 3C 350 INC A ; Cy を変えずに Z=0 にする
31B5 C9 351 RET
31B6 352 ;
31B6 353 ; SPCUT
31B6 354 ;
31B6 355 ; out --- A = スペースを飛ばした後の文字
31B6 356 ; break - F, HL
31B6 357 ;
31B6 358 SPCUT:
31B6 2A D9 31 359 LD HL,(KBPTR)
31B9 360 SPCUT1:
31B9 7E 361 LD A,(HL)
31BA 23 362 INC HL
31BB FE 20 363 CP ' '
31BD 28 FA 364 JR Z,SPCUT1
31BF 22 D9 31 365 LD (KBPTR),HL
31C2 C9 366 RET
31C3 367 ;
31C3 368 ; CAPITAL
31C3 369 ;
31C3 370 ; in --- A = 文字
31C3 371 ; out --- A = 大文字
31C2 372 ; break - F
31C3 373 ;
31C3 374 CAPITAL:
31C3 FE 61 375 CP 'a'
31C5 D8 376 RET C
31C6 FE 7B 377 CP 'z'+1
31C8 D0 378 RET NC
31C9 D6 20 379 SUB 'a'-'A'
31CB C9 380 RET
31CC 381 ;
31CC 382 ; PRTRSET
31CC 383 ;
31CC 384 ; break - F, A
31CC 385 ;
31CC 386 PRTRSET:
31CC 3A DB 31 387 LD A,(PRTRSW)
31CF B7 388 OR A
31D0 C8 389 RET Z
31D1 C3 D9 1F 390 JP #LPTON ; = CALL #LPTON: RET
31D4 391 ;
31D4 392 ; Work Area
31D4 393 ;
31D4 394 WIDMODE: ; 表示桁80桁(1)/40桁(0)
31D4 00 395 DS 1
31D5 396 DEVICE: ; デバイス名
31D5 00 397 DS 1
31D6 398 RECORD: ; R・Wコマンドのレコード省略時
値
31D6 00 00 399 DS 2
31D8 400 WRCBK: ; Wコマンドで省略時値を1戻す(1)
)/戻さない(0)
31D8 00 401 DS 1
31D9 402 KBPTR: ; キーバッファの注目アドレス
31D9 00 00 403 DS 2
31DB 404 PRTRSW: ; 印字ON(1)/OFF(0)
31DB 00 405 DS 1
31DC 406 DUMPOS: ; DUMP ルーチンのオフセット表示
値
31DC 00 407 DS 1
31DD 408 ;
31DD 409 DUMPWK:
31DD 00 00 00 410 DS 18
31E0 00 00 00
31E3 00 00 00
31E6 00 00 00
31E9 00 00 00
31EC 00 00 00
OBJECT CODE END B1EE

```

## ▶ 全機種共通システムインデックス ◀

\*以下のアプリケーションは、基本システムであるS-OS "MACE" またはS-OS "SWORD" がないと動作しませんのでご注意ください。

### 1985年6月号

- 序論 共通化の試み
- 第1部 S-OS "MACE"
- 第2部 Lisp-85インタプリタ
- 第3部 チェックサムプログラム
- 85年7月号
- 第4部 マシン語プログラム開発入門
- 第5部 エディタエンプラZEDA

### 第6部 デバッグツールZAID

- 85年8月号
- 第7部 ゲーム開発パッケージBEMS
- 第8部 ソースジェネレータZING
- 85年9月号
- インタラプト S-OS番外地
- 第9部 マシン語入力ツールMACINTO-S
- 第10部 Lisp-85入門(1)

### ■85年10月号

- 第11部 仮想マシンCAP-X85
- 連載 Lisp-85入門(2)
- 85年11月号
- 連載 Lisp-85入門(3)
- 85年12月号
- 第12部 Prolog-85発表

- 86年1月号  
 第13部 リロケータブルのお話  
 第14部 FM音源サウンドエディタ  
 ■86年2月号  
 第15部 S-OS "SWORD"  
 第16部 Prolog-85入門(1)  
 ■86年3月号  
 第17部 magiFORTH発表  
 連載 Prolog-85入門(2)  
 ■86年4月号  
 第18部 思考ゲームJEWEL  
 第19部 LIFE GAME  
 連載 基礎からのmagiFORTH  
 連載 Prolog-85入門(3)  
 ■86年5月号  
 第20部 スクリーンエディタE-MATE  
 連載 実戦演習magiFORTH  
 ■86年6月号  
 第21部 Z80TRACER  
 第22部 magiFORTH TRACER  
 第23部 ディスクダンプ&エディタ  
 第24部 "SWORD" 2000 QD  
 連載 対話で学ぶmagiFORTH  
 特別付録 PC-8801版S-OS "SWORD"  
 ■86年7月号  
 第25部 FM音源ミュージックシステム  
 付録 FM音源ボードの製作  
 連載 計算力アップのmagiFORTH  
 特別付録 SMC-777版S-OS "SWORD"  
 ■86年8月号  
 第26部 対局五目並べ  
 第27部 MZ-2500版S-OS "SWORD"  
 ■86年9月号  
 第28部 FuzzyBASIC発表  
 連載 明日に向かってmagiFORTH  
 ■86年10月号  
 第29部 ちょっと便利な拡張プログラム  
 第30部 ディスクモニタDREAM  
 第31部 FuzzyBASIC料理法<1>  
 ■86年11月号  
 第32部 バズルゲームHOTTAN  
 第33部 MAZE in MAZE  
 連載 FuzzyBASIC料理法<2>  
 ■86年12月号  
 第34部 CASL & COMET  
 連載 FuzzyBASIC料理法<3>  
 ■87年1月号  
 第35部 マシン語入力ツールMACINTO-C  
 連載 FuzzyBASIC料理法<4>  
 ■87年2月号  
 第36部 アドベンチャーゲームMARMALADE  
 第37部 テキアベ作成ツールCONTEX  
 ■87年3月号  
 第38部 魔法使いはアニメが大好き  
 第39部 アニメーションツールMAGE  
 付録 "SWORD" 再掲載とMAGICの標準化  
 ■87年4月号  
 第40部 INVADER GAME  
 第41部 TANGERINE  
 ■87年5月号  
 第42部 S-OS "SWORD" 変身セット  
 第43部 MZ-700用 "SWORD" をQD対応に  
 ■87年6月号  
 インタラプト コンパイラ物語  
 第44部 FuzzyBASICコンパイラ  
 第45部 エディタアセンブラZEDA-3  
 ■87年7月号  
 第46部 STORY MASTER  
 ■87年8月号  
 第47部 バズルゲーム碁石拾い  
 第48部 漢字出力パッケージJACKWRITE  
 特別付録 FM-7/77版S-OS "SWORD"  
 ■87年9月号  
 第49部 リロケータブル逆アセンブラInside-R  
 特別付録 PC-8001/8801版S-OS "SWORD"  
 ■87年10月号  
 第50部 tiny CORE WARS

- 第51部 FuzzyBASICコンパイラの拡張  
 第52部 Xturbo版S-OS "SWORD"  
 ■87年11月号  
 序論 神話のなかのマイクロコンピュータ  
 付録 S-OSの仲間たち  
 第53部 もうひとつのFuzzyBASIC入門  
 第54部 ファイルアロケータ&ローダ  
 インタラプト S-OSこちら集中治療室  
 第55部 BACK GAMMON  
 ■87年12月号  
 第56部 タートルグラフィックパッケージTURTLE  
 第57部 Xturbo版 "SWORD" アフターケア  
 ラインプリントルーチン  
 特別付録 PASOPIA7版S-OS "SWORD"  
 ■88年1月号  
 第58部 FuzzyBASICコンパイラ・奥村版  
 付録 石上版コンパイラ拡張部の修正  
 ■88年2月号  
 第59部 シューティングゲームELFES  
 ■88年3月号  
 第60部 構造型コンパイラ言語SLANG  
 ■88年4月号  
 第61部 デバッグングツールTRADE  
 第62部 シミュレーションウォーゲームWALRUS  
 ■88年5月号  
 第63部 シューティングゲームELFES II  
 第64部 地底最大の作戦  
 ■88年6月号  
 第65部 構造化言語SLANG入門(1)  
 第66部 Lisp-85用NAMPAシミュレーション  
 ■88年7月号  
 第67部 マルチウィンドウドライバMW-I  
 連載 構造化言語SLANG入門(2)  
 ■88年8月号  
 第68部 マルチウィンドウエディタWINER  
 ■88年9月号  
 第69部 超小型エディタTED-750  
 第70部 アフターケアWINERの拡張  
 ■88年10月号  
 第71部 SLANG用ファイル入出力ライブラリ  
 第72部 シューティングゲームMANKAI  
 ■88年11月号  
 第73部 シューティングゲームELFES IV  
 ■88年12月号  
 第74部 ソースジェネレータSOURCERY  
 ■89年1月号  
 第75部 バズルゲームLAST ONE  
 第76部 ブロックゲームFLICK  
 ■89年2月号  
 第77部 高速エディタアセンブラREDA  
 特別付録 XI版S-OS "SWORD" <再掲載>  
 ■89年3月号  
 第78部 Z80用浮動小数点演算パッケージSOR  
 OBAN  
 ■89年4月号  
 第79部 SLANG用実数演算ライブラリ  
 ■89年5月号  
 第80部 ソースジェネレータRING  
 ■89年6月号  
 第81部 超小型コンパイラTTC  
 ■89年7月号  
 第82部 TTC用バズルゲームTICBAN  
 ■89年8月号  
 第83部 CP/M用ファイルコンバータ  
 ■89年9月号  
 第84部 生物進化シミュレーションBUGS  
 ■89年10月号  
 第85部 小型インタプリタ言語TTI  
 ■89年11月号  
 第86部 TTI用バズルゲームPUSH BON!  
 ■89年12月号  
 第87部 SLANG用リダイレクションライブラリDIO.LIB  
 ■90年1月号  
 第88部 SLANG用ゲームWORM KUN  
 特別付録 再掲載SLANGコンパイラ  
 ■90年2月号  
 第89部 超小型コンパイラTTC++

- 90年3月号  
 第90部 超多機能アセンブラOHM-Z80  
 ■90年4月号  
 第91部 ファジコンビュータシミュレーションMY  
 ■90年5月号  
 第92部 インタプリタ言語STACK  
 ■90年6月号  
 第93部 リロケータブルフォーマットの取り決め  
 第94部 STACK用ゲームSQUASH!  
 第95部 X68000対応S-OS "SWORD"  
 特別付録 PC-286対応S-OS "SWORD"  
 ■90年7月号  
 第96部 リロケータブルアセンブラWZD  
 ■90年8月号  
 第97部 リンカWLK  
 ■90年9月号  
 第98部 BILLIARDS  
 ■90年10月号  
 第99部 ライブラリアンWLB  
 ■90年11月号  
 第100部 タブコード対応エディタEDC-T  
 ■90年12月号  
 第101部 STACKコンパイラ  
 ■91年1月号  
 第102部 ブロックアクションゲームCOLUMNS  
 ■91年2月号  
 第103部 ダイスゲームKISMET  
 ■91年3月号  
 第104部 アクションゲームMUD BALLIN'  
 ■91年4月号  
 第105部 SLANG用カードゲームDOBON  
 ■91年5月号  
 第106部 実数型コンパイラ言語REAL  
 ■91年6月号  
 第107部 Small-C処理系の移植  
 ■91年7月号  
 第108部 REALソースリスト編  
 ■91年8月号  
 第109部 Small-Cライブラリの移植  
 ■91年9月号  
 第110部 SLANG用NEWファイル出力ライブラリ  
 ■91年10月号  
 第111部 Small-C活用講座(初級編)  
 ■91年11月号  
 第112部 Small-C活用講座(応用編)  
 第113部 MORTAL  
 ■91年12月号  
 第114部 Small-C SLANGコンパチ関数  
 ■92年1月号  
 第115部 LINER  
 ■92年2月号  
 第116部 シミュレーションゲームPOLANYI  
 ■92年3月号  
 第117部 カードゲームKLONDIKE  
 ■92年4月号  
 第118部 オプティマイザO80実践Small-C講座(1)  
 ■92年5月号  
 第119部 COMMAND.OBJ実践Small-C講座(2)  
 ■92年6月号  
 第120部 COMMAND.OBJ2実践Small-C講座(3)  
 ■92年7月号  
 第121部 関数リファレンス実践Small-C講座(4)  
 ■92年8月号  
 第122部 ワイルドカード実践Small-C講座(5)  
 第123部 グラフィックライブラリ GRAPH.LIB  
 ■92年9月号  
 第124部 O-EDIT&MODCNV  
 ■92年10月号  
 第125部 SLENDER HUL実践Small-C講座(6)  
 ■92年11月号  
 第126部 EDIT実践Small-C講座(7)  
 ■92年12月号  
 第127部 MAKE実践Small-C講座(8)





# 誤差の少ない三角形自由変形

Shibata Atsushi 柴田 淳

今回は9月号の続きで柴田氏の再登場。7月号の三角形の塗り潰しルーチンを改良して、より正確な三角形の自由変形の実現に挑戦です。9月号で少し予告したアンチエイリアスを導入しています。並行して読むと理解しやすいですよ。

柴田淳(以下Ats): ピアスの穴ってあるじゃないですか。あれってふさがらないんですよね。

琴張春香(以下春): 開けてすぐならふさがるけど、定着しちゃうとね。

琴張護(以下護): そのピアスの穴がどうかしたのでしょうか。

マスター(以下M): まさか、柴田君もピアスをしたくなったとか。

Ats: そんなばかな。いやね、鼻にピアスしてる人っているじゃないですか。そういう取り返しのつかないことをする人がいるんだなあって思ったんですよ。

M: たしかに、耳に穴開ける分には髪の毛なんかで隠れるから目立たないけど、鼻はちょっとアレかなあ。

春: でも、あれって一種のファッションでしょ。

Ats: ファッションだから問題があるんですよ。鼻ピアスなんてそう長くはやってるものではないでしょう。

護: だいたい、ファッション全体が移り変わりやすいものです。

Ats: つまり、ファッションっていうのはクラッシュアンドビルドを大前提にして成り立っているようなものなんです。そのファッションに流されて、一生消えないような傷を自ら体に負わすようなことは、どうかなあと思うんです。

春: うーん、なるほど。そういう意味では、入れ墨なんかも同類よね。

M: あと、日焼けサロンね。街なんかでよく、いかにも日焼けサロンで焼いたふうで、顔とか肩とかシミだらけにしている女性を見かけますよね。

護: ああいうシミは、ちょっとやそつとで

は取れないでしょう。

Ats: まあ、人のことだから関係ないんですけどね。でも、鼻に穴開けちゃった人がジイサンになって、膝に抱いた孫に「ねえ、おじいちゃん、どうしてお鼻に穴が開いているの」とかって聞かれた日には、どんな気持ちがするんだろうなあって、ひとごとながら心配になっちゃうんですよ。



## 誤差の出所

護: ところで、三角形自由変形の依頼者から、変形誤差をどうにかして取り除いてくれないかといわれているのですが。

Ats: ああ、そのことならもうマスターから話を聞いてますよ。一応出来上がったんで、今日持ってきたんですよ。

M: 変形誤差といっても、それほどはなはだしいものじゃなかったような気がしますけど、やっぱりこだわる人はこだわるんですね。

護: こだわりがどうこうということではなくても、やっぱりモーフィングなどに使うことを考えると、一定のクオリティは要求されて当然でしょう。

Ats: そうなんですよね。リアルタイムのシステムで使うならともかく、アニメーション制作などに使うものなら遅くてもきれいなもののほうが重宝がられるんですよ。あんまり遅くても問題でしょうけどね。

春: ところで、三角形の自由変形の誤差ってどうして出るんだったかしら。

護: そういえば、誤差の原因についての話はまだしていなかったのではないのでしょうか。

Ats: じゃあ、今回はそこから始めるとし

## FILE-VI



illustration: T. Takahashi

て、その前に誤差そのものの定義について少々。まず、誤差の一般的な定義を考えると、「同じとされる2つ以上のものを比べ、比べたものの間に差が認められる場合、それが誤差である」というふうになるでしょうか。これが、いちばん基本的な形。

M: 要するに、「間違い=誤差」ってことですよな。

春: でも「いちばん基本的な」なんて、なんか誤差の定義にその先があるようないいまわしね。

護: いまの定義はたしかに一般的ではありますが、本質をついていません。その先があるとしたら、それこそがその本質をついた定義ではないでしょうか。

Ats: おつ、鋭いですね。じゃあ、問題をわかりやすくするために、具体的な数字をコピーする場合の誤差について考えてみましょう。たとえば、1 2 3 という数列を伝言ゲームの要領で伝えていくとしましょう。

春: 伝言ゲームって、文章なんかを人から人に伝えていくゲームね。

Ats: さてA、Bの2チームでその伝言ゲームをやったとして、Aチームの最後の人に伝わったのが1 3 2、Bチームは1 2 0 だったとしましょう。

護: 人から人へと伝えていく間に、数字が変わってしまったのですね。

M: なるほど。これはどちらも間違い、つまり誤差ですよな。

Ats: でも、この2つの答えには本質的な違いがあります。まずAチームのほうは、問題の数列と比べて、使われている数字の種類は同じですよな。

春: 本当だ。でも、Bチームは別の数字が紛れ込んでいるから、こちらのほうが悪い誤

差ね。

護：いや、そうとはいえません。AチームもBチームも、誤差であることには変わりないのです。同じ誤差である以上、優劣は決められないのです。

春：もう、護ちゃんって理屈っぽいんだから。

護：そ、そのようにいわれましても……。

Ats：でも、琴張さんのいっていることは正しいですよ。誤差の優劣より、むしろ注目してほしいのは誤差の仕組みのほうなんですけど。

M：誤差の仕組みというとは？

Ats：いいですか、Aチームの答えは、問題の数列と使われている数字の種類が同じである、つまり「順番が入れ替わっている」誤差なんです。

M：なるほど、その論法でいくと、Bチームの答えは「数字が変わってしまった」誤差ということになるかな。

Ats：そうですね。で、誤差の起こる仕組みというのはコピーするデータの形式と、コピーを行う機構の性格によるんですが、誤差の対処法を考えると、この仕組みを踏まえていなければならないんです。

春：どういうこと？

Ats：Aチームの誤差の原因は、「数字の順番を、ときどき入れ替えて覚えてしまう人が紛れ込んでいる」ことに、たぶんあるんだろうって推測できるでしょう。

M：なるほどね。そうやって原因を特定してから、それを取り除くなりして問題を解決するわけか。

護：ちょっと待ってください。AチームとBチームは、もしかしたら同じ仕組みで起きている誤差だ、という可能性もあります。

Ats：えっ、そうですか？

護：Aチームの解答1 3 2というのは、順番が入れ替わったのではなくて「たまたま2番目と3番目の数字がそれぞれ3と2に変わった」という解釈も可能なのではないのでしょうか。

Ats：うーん、そうか。それを確かめるには、あと何回か伝言ゲームをしてみるしかないですね。たとえば、1から9まで全部をひとつずつ使った数列を伝言するとか。

春：数字が変わっているのなら、答えのなかに同じ数字が現れるだろうから、たまたま違う数字と入れ替わったのかどうかを調

べるためには、それを何回か繰り返せばいいのね。



## ディスプレイ上の誤差

Ats：では伝言ゲームの例はこのくらいにして、今度は話をもっと進めるために、コンピュータのビットマップディスプレイ上で画像を変形コピーする場合を考えてみましょう。

M：ただコピーする場合じゃなくて、変形もしちゃうんですか？

Ats：いえね、ただのコピーの場合と変形コピーの場合だと、誤差の定義が大きく変わっちゃうんです。つまり、変形の操作自体がコピー元を、つまりオリジナルを変えてしまうじゃないですか。

護：変形によって変わった要素は、誤差のうちに含めないということですね。

Ats：だいいちディスプレイ上の色というのは、コンピュータでは数値として扱われるわけですから、ただのコピーをするときに起こる問題というのは伝言ゲームと変わらないのですよ。

春：数字をひとつずつ、たくさん受け渡せばいいんだものね。

護：コンピュータの場合はよほどのことがない限りデータは元のまま受け渡されますから、変形元と変形先の点の対応だけをしつかり取っておけば、問題は発生しないでしょう。

Ats：さて、ここでも問題を単純にするために、ある長方形内の画像を横幅が半分の長方形のなかに押し込む変形をするのでしょうか。

春：いちばんオーソドックスな方法は、元の画像のドットをひとつ飛ばしに、変形先にコピーしていくという方法かしら。

M：横幅が半分ってことは、変形先の横のドット数が半分ってことだから、変形の際に起こる「データの欠損」はどうしても避けられないですね。

Ats：そうなんです。画像変形の際、いちばん難しいのがそのところなんだよな。で、どうするかなんですけど。

護：そうですね、変形元の隣り合った2ドットの色の中間を取り、変形先にコピーしていけばより元画像に近い変形画像が得られるのではないのでしょうか。

春：なるほどね。

Ats：誤差というのは、難しい言葉を使えば「エントロピーの増大」、つまり「コピーからオリジナルを再生しようとするときの困難が増すこと」と定義されるんです。そういう意味において、データの欠損が問題になってくる。

M：でもビットマップの画像の場合、変形によってどうしてもデータを削らなければならない場合が出てくるじゃないですか。

春：しょうがないから、そこでさっき護ちゃんがいったみたいの方法を取るわけね。

Ats：このように、細かすぎて本来ならドット中に再現されない情報をにじませて表示する手法を、アンチエイリアスなんて呼んだりします。

M：そういえばモーフィング実験のとき、アンチエイリアスを使えば三角形自由変形の誤差がなくなるみたいなことをいってませんでしたっけ？

Ats：まあ、それだけじゃ誤差は完全にはなくなりませんがね。それでは次に、三角形の自由変形にそのアンチエイリアスを適用する方法などに触れていきましょう。



## アンチエイリアスの実際

Ats：三角形の自由変形を行うとき、たとえば変形元の三角形に比べて、変形先の面積がとても小さい場合、先ほどの「データの欠損」が大きな誤差を生み出します。

春：で、そこにアンチエイリアスを導入するのね。

Ats：ただ、整数だけを使って変形させようとすると、いろいろ問題が出てくるんですよ。

護：実数を使って変形するならそうでもないのでしょうか。

Ats：ところで、三角形の自由変形の方法って覚えていますか。

M：どんなでしたっけ？

春：わたし、そのときいなかったからわからない。

護：覚えがないですね。

Ats：……まあいいです。とりあえず図1を見てください。

春：図1のAのほうね。

Ats：簡単にアルゴリズムを解説すると、まず変形先の三角形をラスターキャンデ



埋めていき、埋める色は変形元の対応するドットから得る、という感じかな。

M:すると、変形元に比べて変形先の三角形の面積が小さいときは、元の三角形の色を間引いて走査することになります。つまり、間引かれた部分の色の情報が欠落するんですね。

Ats:そうなんです。これを解決するためには、間引く部分の色をどうにかして変形先の1ドットに集めればいいんですが、その方法を図解したのが、図1のBです。

春:この図はどう見るの?

Ats:いいですか、この方法の基本は、ドットに注目するのではなくて、変形先のラスタースタート、対応する変形元の三角形の「弦」に注目するところにあります。

M:弦というのはつまり、三角形の2つの辺を結んだ線、ということですね。

護:ラスタースタートと弦に注目するとはどういうことでしょう。

Ats:図1のBの例でいくと、変形先の1ラスタースタートに押し込めるべき部分というのは、四角形の領域というのはわかりますよね。

春:それはわかるけど、じゃあその四角形はどうやって求めるの?

Ats:まず、目的の四角形の1辺には、ラスタースタートに対応する変形元の弦が必ず含まれます。また、2辺は三角形の辺上の線分だということがわかると思いますけど。

護:なるほど。これで3つの辺が特定できそうですが、あと1辺が決まらなければ四角形にはなりません。

Ats:で、ここがポイントなんですけど、残りの1辺というのは、「次のラスタースタートに対応する弦の直前の弦」になるんです。

M:でも、その四角形を特定したあとはどうするんですか。

Ats:あとは、この四角形を対応するラスタースタートに押し込んでやるだけです。

M:だから、どんな方法でラスタースタートに押し込むか聞いているんじゃないですか。

Ats:そんなに難しいことないですよ。要領としては、四角形の自由変形と同じです。

護:変形元を先ほどの四角形に、変形先を目的のラスタースタートにすればいいのですね。

M:そうか、横1本の線も、一種の四角形と見えないことないものなあ。

Ats:ただし、変形先の四角形の「高さ」を考えなくていいから、その分処理は簡単に

なりますけどね。

M:でも、この処理も実数を使わずにやってるんですね。

Ats:そうですよ。基本は以前の三角形自由変形のときにやった「線分上を指定回数で動かす」というアレなんです。

M:でも、その方法を使ってもやっぱり走査する点は間引くことになっちゃうじゃないですか。

Ats:そんなことないですよ。指定する回数に、「変形先、元のどちらか面積の大きなほうをくまなく走査するような値」を取ればいいんですからね。

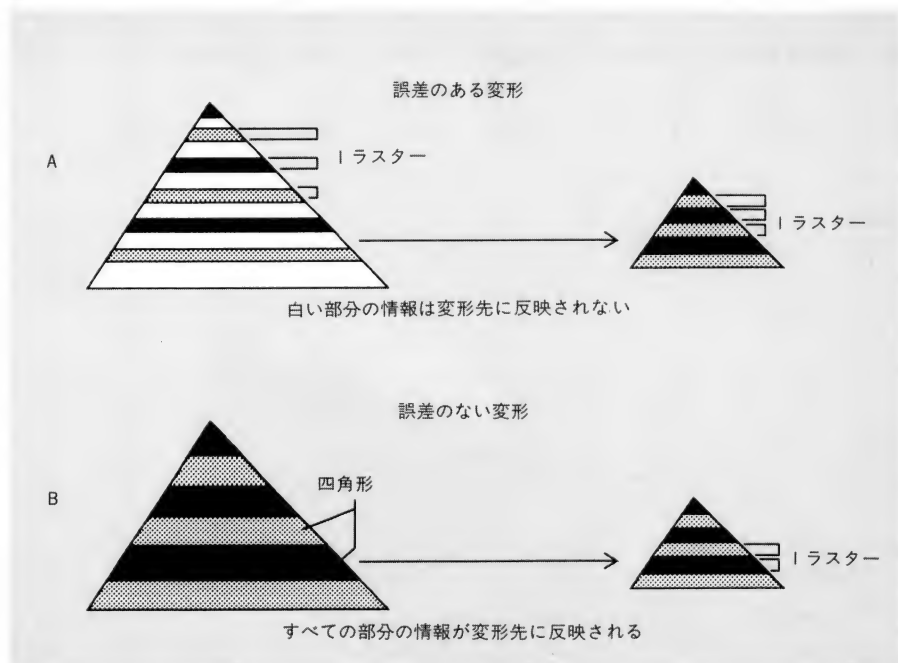
M:あともうひとつ。四角形をラスタースタートに押し込む処理をしているとき、ラスタースタートの同一点に複数の色が重なるから、重なった色の中間色を取るんでしたよね。

護:中間色の取り方なら私にもわかります。2つの色をRGBの要素に分けて、要素ごとに値の平均を取ればいいのです。

M:いや、そういうことじゃなくてね、中間色を取る時点で割り算をするわけですから、そこにもやはり誤差が出るんじゃないかと思って。割り算も、整数でやっているんじゃないしょう。

Ats:たしかに、誤差は出ます。2で割るから、ビットを右にシフトすることになるんですけど、そうするとシフトする前に立っていた第0ビットが計算結果に反映されないということになりますね。

図1



護:この誤差は取り除くことはできないでしようか。

Ats:この誤差の原因には、ターゲットマシンのグラフィック機能の限界という動かし難い要素がからんでいるので、これが大きな障壁ですよ。誤差拡散法なんていうおもしろい方法もありますが、ラスタースキャンのアルゴリズムにコイツをもぐり込ませられるかは、微妙なところだと思います。

M:まあ、変形を何度も繰り返すのなら話は別ですが、元画像からの変形を1回しかしないのならそれほど気にならない誤差かもしれないですね。

## ドットがズレる

Ats:さて、色が抜け落ちる誤差ほど深刻ではありませんが、もうひとつ取り除いた誤差があります。

M:もうひとつの誤差というと?

Ats:以前テキスト三角形塗り潰しをやったとき「太った三角形とやせた三角形」というのをやりましたよね。

M:どんなのでしたっけ?

Ats:そうくると思いましたよ、まったく。こちら図2を用意しましたので、それを見てください。

護:ラインで三角形の辺を描いてそのなかを塗り潰したものと、三角形塗り潰しルー



チンで描いたものとは、後者のほうが「やせた三角形」になってしまうというものです。

**Ats:** 以前の自由変形ルーチンだと、変形先の三角形の輪郭をなぞるのに、この「やせた三角形」を出力する方法を取っていたんです。一方、変形元の三角形の輪郭はどうかというと、こちらは「太った三角形」を出力する方法だった。

**護:** つまり、変形先と変形元の座標の対応が正しく取れていないのですね。

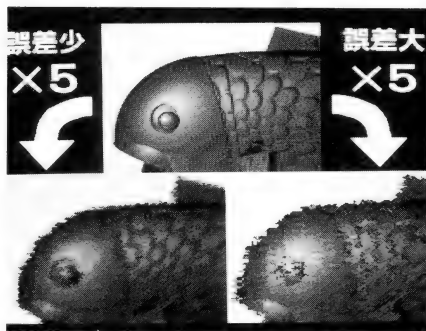
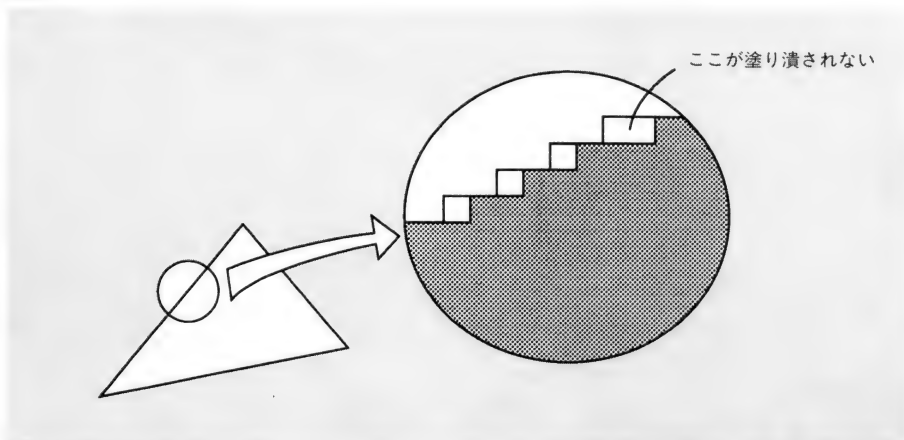
**Ats:** そうなんです。それで変形結果がズレるというか、歪んだようになってしまいうんです。

**M:** ちょうど、伝言ゲームの例のAチームの答えの誤差の原因にタイプが2つあるのと同じですね。

**護:** つまり誤差のタイプが違うから、先ほどのアンチエイリアスではこの誤差は吸収できないということでしょうか。

**Ats:** 輪郭を抽出する方法をどちらかに統一すればいいんです、解決方法としてはね。で、どちらに統一するかというと、これはやはり「太った三角形」のほうがいいと。

図2



誤差大のほうは、目が潰れている

**M:** 「やせた三角形」に統一すると、変形先の三角形が埋まりきらなくなりますからね。

**Ats:** このへんのことは、7月号の三角形塗り潰しルーチンでやったことなんで、詳しくは触れません。で、新しい自由変形ルーチンがこれなんですけど (リスト参照)。

**M:** あれ、Cで書いてありますね。いつもアセンブラのソースで書いてくるのに。

**Ats:** すいません。今回は時間がなかったんで。

**護:** 手抜きですね。

**Ats:** なにってんですか。アセンブラで書くとデバッグが大変なんです。

**護:** じゃあ、パワーダウンだ。

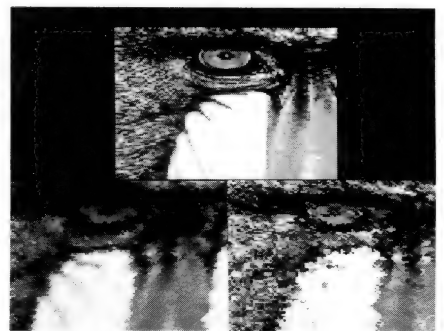
**Ats:** いちいちつかかる人だなあ。

**M:** まあまあお2人ともそんなにあつくないで。で、プログラミングのポイントかなんかないんですか？

**Ats:** 繰り返しになりますが、処理はすべて整数を使っています。だからアセンブラで書いたみたいに速いか、ということでもないところが我ながら情けないところなんですけどね。

**M:** どこで時間を食っているんでしょう。

**Ats:** たぶん、中間色を取る部分がいちば



マンドリルの顔がずいぶん崩れている

ん重いんでしょう。シフトとORしかしていないから、アセンブラソースに書き換えればけっこう速くなると思いますけどね。

**護:** アルゴリズムで特別なことをしている部分などはないのですか。

**Ats:** アルゴリズムはいままで使ってきたものをCのソースに落としてあるだけです。だから逆に、これをアセンブラソースに書き換えるのもそれほど大変じゃないはずなんですけどね。

**護:** では書き換えればいいじゃないですか。

**Ats:** また、ひとごとだと思って。

**護:** ひとごとではないですか。

**M:** あれ、今回はサンプルみたいなのはいいんですか？

**Ats:** 以前の自由変形ルーチンと差し替えばいいので、サンプルは特に用意しませんでした。ただそれじゃああんまりだから、どの程度誤差がなくなっているかわかるような画面写真を用意しました。あらかじめいっておきますが、まだ完全には誤差は取り除けていないですよ。

**護:** なるほど。ところどころあやしい部分があります。

**Ats:** いや、でも今回自由変形の誤差を取り除こうとプログラミングしてみて、はじめに思ってた以上に難しいんでびっくりするやらあせるやら、とにかく大変でしたよ。

**M:** そういうことを考えると、微細加工技術とか、いったいどうやって誤差を吸収しているのか見当もつきませんね。

**Ats:** あと、宇宙探査とかね。木星より遠い惑星を探索するときなんか、小惑星体の微小重力とかが影響するはずじゃないですか。まさか1つひとつの小惑星の軌道を考慮に入れて軌道計算してるんじゃないでしょうし。

**M:** 探査期間も数年単位ですから、量子論的な誤差も影響してくるんじゃないでしょ



うか。

護：聞いた話によると、あれはむしろ軌道誤差を出すように航行しているのだそうです。ただし、確率的に誤差が最小限に収束するような範囲で、ということらしいです。

Ats：へえ、そこまでいくともうついていけないなあ。宇宙開発やっている人たちって、ほんとうは宇宙人かなにかなんじゃないでしょうかねえ。僕にやとても信じられないや。

M：あれ、ところで春香さんがいませんね。話に夢中になってたんで、いついなくなったのか気づかなかったけれど。

Ats：あつ、机の上に置き手紙が。「護ちゃんのパカ！」って書きなぐってありますよ。

護：がーん！

M：まさか、あんまりややこしい話をするんで、嫌になって帰っちゃったとか……。

Ats：ちょっと琴張さん、家に電話したほうがいいんじゃないですか。

護：は、春香さん、そんな……。

M：なんか、あまりのショックで放心状態みたいですね。

Ats：そんな、奥さんに嫌われたくらいでそんなに思い詰めなくてもいいじゃないですか。

護：わ、私がバカだなんて……。

M：……。なんか違う意味でショック受けてるみたいですよ。

Ats：この夫婦は、案外これでお似合いなのかもしれない。

つづく

## リスト

```
1: /*
2: 誤差の少ない三角形自由変形
3:
4: SEP.17th.1993 (ats)
5: */
6: #include "stdio.h"
7: #include "basic.h"
8: #include "graph.h"
9:
10: struct POINTS {
11:     int x1,y1,x2,y2,x3,y3;
12:     struct PRMS {
13:         int x,y,dx,dxx,dyy,dir,
14:         dir2,dy,vx,vy,dv;
15:     };
16:     int x[2][3],y[2][3],xx[4],yy[4];
17:     int pre[4][512],aft[4][512];
18:     int c_buf[512];
19: void trtrfm( pre_p,aft_p )
20: struct POINTS *aft_p,*pre_p;
21: {
22:     int i,j,k,l,m,d[3];
23:     struct PRMS p[3];
24:     char cc;
25:     /* 座標を配列に */
26:     x[0][0] = (*aft_p).x1;
27:     y[0][0] = (*aft_p).y1;
28:     x[0][1] = (*aft_p).x2;
29:     y[0][1] = (*aft_p).y2;
30:     x[0][2] = (*aft_p).x3;
31:     y[0][2] = (*aft_p).y3;
32:     x[1][0] = (*pre_p).x1;
33:     y[1][0] = (*pre_p).y1;
34:     x[1][1] = (*pre_p).x2;
35:     y[1][1] = (*pre_p).y2;
36:     x[1][2] = (*pre_p).x3;
37:     y[1][2] = (*pre_p).y3;
38:     /* 座標の並べ替え */
39:     swap( 0,1 );
40:     swap( 1,2 );
41:     swap( 0,1 );
42:     calc_parm( 0,0,1,&p[0] );
43:     p[0].x += p[0].dxx/2;
44:     calc_parm( 0,0,2,&p[1] );
45:     p[1].x += p[1].dxx/2;
46:     calc_parm( 0,1,2,&p[2] );
47:     if( p[0].dir == 1 )
48:         p[2].x += p[2].dxx/2;
49:     else
50:         p[2].x -= p[2].dxx/2;
51:     /* 描画先の点の生成 */
52:     i = 0;
53:     if( p[0].y != y[0][1] )
54:     {
55:         while( p[0].y <= y[0][1] )
56:         {
57:             aft[0][i] = p[0].x;
58:             aft[1][i] = p[0].y;
59:             aft[2][i] = p[1].x;
60:             aft[3][i] = p[1].y;
61:             fl_edg( &p[0] );
62:             fl_edg( &p[1] );
63:             i++;
64:         }
65:         fl_edg( &p[2] );
66:     }
67:     while( p[1].y <= y[0][2] )
68:     {
69:         aft[0][i] = p[2].x;
70:         aft[1][i] = p[2].y;
71:         aft[2][i] = p[1].x;
72:         aft[3][i] = p[1].y;
73:         fl_edg( &p[2] );
74:         fl_edg( &p[1] );
75:         i++;
76:     }
77:     calc_parm2( 1,0,1,y[0][1]-y[0][0],&p[0] );
78:     calc_parm2( 1,0,2,y[0][2]-y[0][0],&p[1] );
79:     calc_parm2( 1,1,2,y[0][2]-y[0][1]+1,&p[2] );
80:     /* 変形元の点の生成 */
81:     k = 0;
82:     j = y[0][1]-y[0][0]+1;
83:     if( p[0].dv != 0 )
84:         for( i = 0; i != j; i++ )
85:         {
86:             pre[0][k] = p[0].x;
87:             pre[1][k] = p[0].y;
88:             pre[2][k] = p[1].x;
89:             pre[3][k] = p[1].y;
90:             fl_edg2( &p[0] );
91:             fl_edg2( &p[1] );
92:             k++;
93:         }
94:         fl_edg2( &p[2] );
95:         j = y[0][2]-y[0][1];
96:         for( i = 0; i != j; i++ )
97:         {
98:             pre[0][k] = p[2].x;
99:             pre[1][k] = p[2].y;
100:             pre[2][k] = p[1].x;
101:             pre[3][k] = p[1].y;
102:             fl_edg2( &p[2] );
103:             fl_edg2( &p[1] );
104:             k++;
105:         }
106:         d[0] = abs(y[1][1]-y[1][0]);
107:         if( d[0] < abs(x[1][1]-x[1][0]) )
108:             d[0] = abs(x[1][1]-x[1][0]);
109:         d[1] = abs(y[1][2]-y[1][0]);
110:         if( d[1] < abs(x[1][2]-x[1][0]) )
111:             d[1] = abs(x[1][2]-x[1][0]);
112:         d[2] = abs(y[1][2]-y[1][1]);
113:         if( d[2] < abs(x[1][2]-x[1][1]) )
114:             d[2] = abs(x[1][2]-x[1][1]);
115:         calc_parm2( 1,0,1,d[0],&p[0] );
116:         calc_parm2( 1,0,2,d[1],&p[1] );
117:         calc_parm2( 1,1,2,d[2],&p[2] );
118:         i = 0;
119:         xx[0] = pre[0][0];
120:         yy[0] = pre[1][0];
121:         xx[1] = pre[2][0];
122:         yy[1] = pre[3][0];
123:         xx[2] = pre[0][0];
124:         yy[2] = pre[1][0];
125:         xx[3] = pre[2][0];
126:         yy[3] = pre[3][0];
127:         if( d[0] != 0 )
128:         {
129:             while( p[0].x != x[1][1] &&
130:                   p[0].y != y[1][1] )
131:             {
132:                 while( p[0].x != pre[0][i] &&
133:                       p[0].y != pre[1][i] )
134:                 {
135:                     xx[2] = p[0].x;
136:                     yy[2] = p[0].y;
137:                     fl_edg2( &p[0] );
138:                 }
139:                 while( p[1].x != pre[2][i] &&
140:                       p[1].y != pre[3][i] )
141:                 {
142:                     yy[3] = p[1].y;
143:                     xx[3] = p[1].x;
144:                     fl_edg2( &p[1] );
145:                 }
146:                 l_att( i );
147:                 xx[0] = pre[0][i];
148:                 yy[0] = pre[1][i];
149:                 xx[1] = pre[2][i];
150:                 yy[1] = pre[3][i];
151:                 i++;
152:             }
153:         }
154:         while( i < k )
155:         {
156:             while( p[2].x != pre[0][i] &&
157:                   p[2].y != pre[1][i] )
158:             {
159:                 xx[2] = p[2].x;
160:                 yy[2] = p[2].y;
161:                 fl_edg2( &p[2] );
162:             }
163:             while( p[1].x != pre[2][i] &&
164:                   p[1].y != pre[3][i] )
165:             {
166:                 xx[3] = p[1].x;
```

```

167:         yy[3] = p[1].y;
168:         fl_edg2( &p[1] );
169:     }
170:     l_att( i );
171:     xx[0] = pre[0][i];
172:     yy[0] = pre[1][i];
173:     xx[1] = pre[2][i];
174:     yy[1] = pre[3][i];
175:     i++;
176: }
177: }
178: void swap( n1,n2 )
179: int n1,n2;
180: {
181:     int tmp;
182:     if( y[0][n1] > y[0][n2] )
183:     {
184:         tmp = x[1][n2];
185:         x[1][n2] = x[1][n1];
186:         x[1][n1] = tmp;
187:         tmp = y[1][n2];
188:         y[1][n2] = y[1][n1];
189:         y[1][n1] = tmp;
190:         tmp = x[0][n2];
191:         x[0][n2] = x[0][n1];
192:         x[0][n1] = tmp;
193:         tmp = y[0][n2];
194:         y[0][n2] = y[0][n1];
195:         y[0][n1] = tmp;
196:     }
197: }
198: void calc_parm( n0,n1,n2,prm )
199: int n0,n1,n2;
200: struct PRMS *prm;
201: {
202:     (*prm).x = x[n0][n1];
203:     (*prm).y = y[n0][n1];
204:     (*prm).dy = y[n0][n2]-y[n0][n1];
205:     if( (*prm).dy != 0 )
206:     {
207:         (*prm).dxx = (x[n0][n2]-x[n0][n1])/(*prm).dy;
208:         (*prm).dx =
209:         abs((x[n0][n2]-x[n0][n1])-(*)prm).dxx*(*)prm).dy);
210:         (*prm).dir = sgn( x[n0][n2] - x[n0][n1] );
211:     }
212:     (*prm).vx = (*prm).dy/2;
213:     (*prm).vy = 0;
214: }
215: void calc_parm2( n0,n1,n2,dv,prm )
216: int n0,n1,n2,dv;
217: struct PRMS *prm;
218: {
219:     (*prm).x = x[n0][n1];
220:     (*prm).y = y[n0][n1];
221:     if( dv != 0 )
222:     {
223:         (*prm).dxx = (x[n0][n2]-x[n0][n1])/dv;
224:         (*prm).dyy = (y[n0][n2]-y[n0][n1])/dv;
225:         (*prm).dx = abs((x[n0][n2]-x[n0][n1])-(*)prm).dxx*dv);
226:         (*prm).dy = abs((y[n0][n2]-y[n0][n1])-(*)prm).dyy*dv);
227:         (*prm).dir = sgn( x[n0][n2] - x[n0][n1] );
228:         (*prm).dir2 = sgn( y[n0][n2] - y[n0][n1] );
229:     }
230:     (*prm).vx = dv/2;
231:     (*prm).vy = dv/2;
232:     (*prm).dv = dv;
233: }
234: void fl_edg( prm )
235: struct PRMS *prm;
236: {
237:     (*prm).y++;
238:     (*prm).x += (*prm).dxx;
239:     (*prm).vx += (*prm).dx;
240:     if( (*prm).vx >= (*prm).dy )
241:     {
242:         (*prm).x += (*prm).dir;
243:         (*prm).vx -= (*prm).dy;
244:     }
245: }
246: void fl_edg2( prm )
247: struct PRMS *prm;
248: {
249:     (*prm).x += (*prm).dxx;
250:     (*prm).vx += (*prm).dx;
251:     if( (*prm).vx >= (*prm).dv )
252:     {
253:         (*prm).x += (*prm).dir;
254:         (*prm).vx -= (*prm).dv;
255:     }
256:     (*prm).y += (*prm).dyy;
257:     (*prm).vy += (*prm).dy;
258:     if( (*prm).vy >= (*prm).dv )
259:     {
260:         (*prm).y += (*prm).dir2;
261:         (*prm).vy -= (*prm).dv;
262:     }
263: }
264: void l_att( n )
265: int n;
266: {
267:     int i,j,k,l;
268:     struct PRMS p[2];
269:     j = abs(aft[2][n]-aft[0][n]);
270:     for( i = 0; i != j; i++ )
271:         c_buf[i] = -1;
272:     k = abs(xx[2]-xx[0]);
273:     if( k < abs(yy[2]-yy[0]) )
274:         k = abs(yy[2]-yy[0]);
275:     if( k < abs(xx[3]-xx[1]) )
276:         k = abs(xx[3]-xx[1]);

```

```

277:     if( k < abs(yy[3]-yy[1]) )
278:         k = abs(yy[3]-yy[1]);
279:     if( k == 0 )
280:     {
281:         la_sub( xx[0],yy[0],xx[2],yy[2],j,n );
282:     }
283:     else
284:     {
285:         k *= 2;
286:         calc_parm3( 2,0,k,&p[0] );
287:         calc_parm3( 3,1,k,&p[1] );
288:         for( i = 0; i != k; i++ )
289:         {
290:             la_sub( p[0].x,p[0].y,p[1].x,p[1].y,j,n );
291:             fl_edg2( &p[0] );
292:             fl_edg2( &p[1] );
293:         }
294:     }
295:     if( aft[2][n] > aft[0][n] )
296:         for( i = aft[0][n]; i < aft[2][n]; i++ )
297:             pset( i,aft[1][n],c_buf[i-aft[0][n]] );
298:     if( aft[2][n] < aft[0][n] )
299:         for( i = aft[0][n]; i > aft[2][n]; i-- )
300:             pset( i,aft[1][n],c_buf[i-aft[2][n]] );
301: }
302: void calc_parm3( n1,n2,dv,prm )
303: int n1,n2,dv;
304: struct PRMS *prm;
305: {
306:     (*prm).x = xx[n1];
307:     (*prm).y = yy[n1];
308:     if( dv != 0 )
309:     {
310:         (*prm).dxx = (xx[n2]-xx[n1])/dv;
311:         (*prm).dyy = (yy[n2]-yy[n1])/dv;
312:         (*prm).dx = abs((xx[n2]-xx[n1])-(*)prm).dxx*dv);
313:         (*prm).dy = abs((yy[n2]-yy[n1])-(*)prm).dyy*dv);
314:         (*prm).dir = sgn( xx[n2] - xx[n1] );
315:         (*prm).dir2 = sgn( yy[n2] - yy[n1] );
316:     }
317:     (*prm).vx = dv/2;
318:     (*prm).vy = dv/2;
319:     (*prm).dv = dv;
320: }
321: void la_sub( x1,y1,x2,y2,t,n )
322: int x1,y1,x2,y2,t,n;
323: {
324:     int a,b,i;
325:     register int j,k,c1,c2,c3;
326:     struct PRMS p,s;
327:     a = abs(x2-x1);
328:     if( a < abs(y2-y1) )
329:         a = abs(y2-y1);
330:     b = t;
331:     if( a > b )
332:         b = a;
333:     if( b == 0 )
334:     {
335:         c_buf[0] = point( pre[0][n],pre[1][n] );
336:         return;
337:     }
338:     b++;
339:     p.x = x1;
340:     p.y = y1;
341:     p.dxx = (x2-x1)/b;
342:     p.dyy = (y2-y1)/b;
343:     p.dx = abs((x2-x1)-p.dxx*b);
344:     p.dy = abs((y2-y1)-p.dyy*b);
345:     p.dir = sgn(x2-x1);
346:     p.dir2 = sgn(y2-y1);
347:     p.vx = b/2;
348:     p.vy = b/2;
349:     p.dv = b;
350:     s.dxx = abs(aft[2][n]-aft[0][n])/b;
351:     s.dx = abs((aft[2][n]-aft[0][n])-s.dxx*b);
352:     s.dir = sgn(aft[2][n]-aft[0][n]);
353:     s.vx = b/2;
354:     s.dv = b;
355:     s.x = 0;
356:     if( s.dir == -1 )
357:         s.x = abs(aft[2][n]-aft[0][n]);
358:     for( i = 0; i <= b; i++ )
359:     {
360:         if( c_buf[s.x] == -1 )
361:             c_buf[s.x] = point(p.x,p.y);
362:         else
363:         {
364:             c3 = 0;
365:             c1 = c_buf[s.x];
366:             c2 = point( p.x,p.y );
367:             if( c1 != c2 )
368:             {
369:                 k = (c1 & 0xf800) >> 11;
370:                 j = (c2 & 0xf800) >> 11;
371:                 k = (k+j)>>1;
372:                 c3 = c3 | ( k << 11 );
373:                 k = ( c1 & 0x07c0 ) >> 6;
374:                 j = ( c2 & 0x07c0 ) >> 6;
375:                 k = (k+j)>>1;
376:                 c3 = c3 | ( k << 6 );
377:                 k = ( c1 & 0x003e ) >> 1;
378:                 j = ( c2 & 0x003e ) >> 1;
379:                 k = (k+j)>>1;
380:                 c_buf[s.x] = c3 | ( k << 1 );
381:             }
382:         }
383:         fl_edg2( &p );
384:         fl_edg2( &s );
385:     }
386: }

```



# 進化する目標を追い続けて

Ishibumi Akira

伊瀬見 あきら

究極を目指したジョイスティックもひとまず完成となりました。今回は実際の工程を進めるうえで注意すべきことや、ジョイスティックのメンテナンスなどについて解説します。

また3カ月あとになると思われたこの連載も、奇跡の3回目を迎えました。頼りなく無計画でありながら、X680x0対応の究極のジョイスティックを目指し、ついにここまでやってくることができました。今回は、紙の上ではわからなかった衝撃の事実や、実際に作業して身についたノウハウなどを中心に、ジョイスティックの未来像といった領域まで迫ってみたいと思います。では今月も、究極への一步を踏み出すことにしましょう。

## ▶わかってはいるけど、やっぱり復習◀

今回初めてこの連載の存在に気づいた人や、先月までの話をキレイサッパリ忘れてしまった人に、これまでの経過を説明しておきましょう。

ことの発端は、使い道のなくなったファミコン用のジョイスティックをなんとかX680x0用に転用してほしいという、悪魔の餌をSLASH横内氏が持ち込んだことにあります。

出された注文の必要十分条件を満たす程度の、コネクタを交換して線をつなぎ替えてハイ終わりというのではやっているほうとしても面白くないので、より高度で高尚な目標を立てることにしました。

とりあえずの目標としたのは、

- ・3つボタンにする
- ・ボタンごとに好きな機能を割り当て可能
- ・ボタンごとに連射が選択できる
- ・連射の同期/非同期が選択できる
- ・FM TOWNSのSELECT/STARTボタンに対応

という5つの拡張機能で、これによって現在想定されるあらゆるX680x0ゲームに対応可能な究極のジョイスティックの完成を

目指しました。

そこで、追加のボタン用の穴を開ける道具として、ホールソウという穴開け用のドリル刃を購入し、ボタンの機能を変更可能にする回路の設計を終了したのが先月までのあらすじです。

このほかに必要な基礎知識としては、ジョイスティックの入力端子は0V(GND)につながっているときに入力があるとみなし、なにもつながっていないときは+5Vがつながっているのと等価という電気的特性を持っているということがあります。

通常はスイッチを通して、入力端子を0Vにつなぐかつつながないかで信号を制御しますが、このジョイスティック計画ではICの出力の+5Vや0Vを直接つなぐことで、連射信号やボタン配列の信号をX680x0本体に送り込んでいます。

あとひとつだけ訂正させてもらおうと、FM TOWNSの純正パッドにあるSELECTとSTARTのボタンはジョイスティックの上下を同時にオンにするとSELECTで、左右を同時にオンにすればSTARTになります。先月は逆に書いてしまったので混乱を招いてしまったかもしれません。海よりも深く反省しています。ごめんなさい。

## ▶男はパワーで穴を開ける◀

先月の写真ではすでに穴が開いていましたが、実際、ホールソウを使つての穴開けは簡単です。ただ取り返しのつかない作業なので、穴を開ける位置は慎重に決定してください。くれぐれも、開いた穴にボタンを付けたらボタンがケースの底にぶつかって蓋が閉まらないなんてことにならないように、念には念を入れてください。ボタン追加の場合は穴開けの前に周囲のパーツや

配線を除去しておくことも忘れずに。

ホールソウをつける電動ドリルは普通の日曜大工に使う程度で大丈夫ですが、正確さを期す意味で普通のドリル刃で5mm程度の穴を中心に開けてからホールソウで削り貫くようにすれば完璧です。

ホールソウにも芯になるドリル刃が付いていますが、意外と太いのでmm単位の正確さを要求される作業では念を入れたほうが利口です。

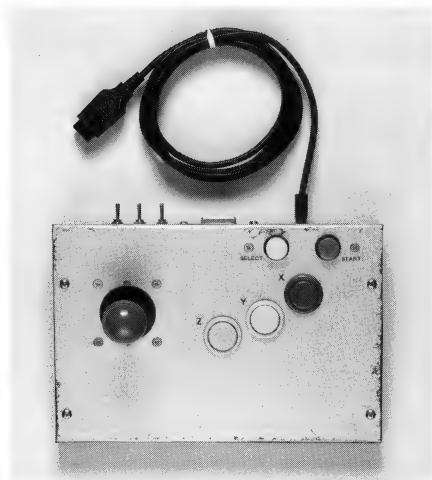
あと、ホールソウでガリガリやっているときには随時水で冷却しながらやるのを忘れないようにしましょう。冷やしていても摩擦熱でかなりの熱を発生しますので。削り貫かれたばかりのドーナツのような破片を無警戒に触るようなこともやってはいけません。

これを先月比較した手動式のネジで挟んで削り貫くホールパンチで行うと、異常なほど面倒でさらに腕力が要求され仕上がりも若干雑になります。結果的にはホールソウのほうが仕上げの安定感や労力の面から見て非常に有利だといえます。なにはともあれ、穴はこれでバッチリ美しいものが開きました。

## ▶完成型を予想しよう◀

3つ目のボタンの穴が開いたので、次はその制御部分を作らなくてはなりません。連射とボタン配列の自由化を可能にする回路の設計は先月で終わっているの、あとは怒涛のハンダ付けかという、そうではありません。いちばんの難題であるスイッチや追加回路のケース内部のレイアウトをやらなくてははいけません。

基板は比較的メジャーなICB-93というタイプの汎用基板を使いました。これをジ

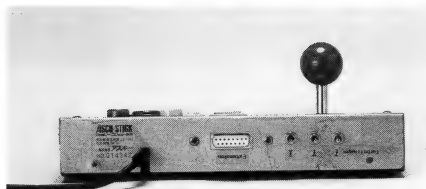
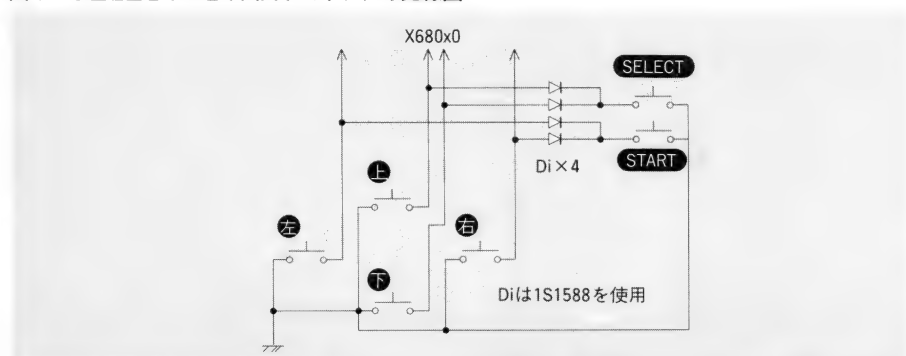


ヨイスティックのケースの中にぶつからないように収める位置を探したところ、右側手前しかありませんでした。また回路の構造上、ボタン配列のディップスイッチや連射速度調整のボリュームを操作するアクセス用の穴も開ける必要があったので、ケースの底に穴を開け、基板の部品面を底に向けることで解決しました。基板の位置が決まったのでアクセス用の穴と基板固定用のネジ穴を開けました。

連射の切り替えスイッチは本来は背面のボタンと同じ側にしたかったのですが、SELECT/STARTボタンに邪魔されたので断念し、スティックのある側の背面に3つのスイッチ用の穴を用意しました。

こうして実情に合わせ、随時部品の位置を決めていながら穴を開けました(あまりほめられた方法ではありませんが、大量に生産する必要がないので十分有効手段だといえます)。ちなみに基板のアクセス用の楕円の穴はホールソーで円を2つ並べて開け、途中をハンドニプラで切り開いて穴をつなげたので見栄えがずいぶんよくなりました。

図1 SELECTとSTARTスイッチの配線図

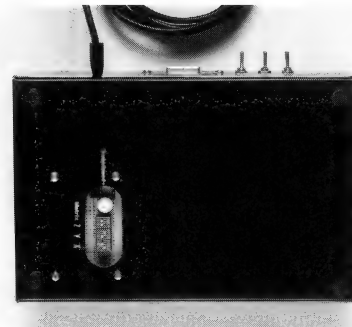


また、なにかケース自体に塗装やデコレーションを施すならば、穴開けなどの加工が終了したこの時点で作業をしておく必要があります。

今回はボタンの穴を増設するときに各種の部品を外した関係上、フロントパネルの化粧シールを剥がす必要がありました。それによって当初の状態に比べてずいぶん無骨な代物に変わってしまったのは否定できませんが、特に代替のデコレーションなどには行いませんでした。計画的に塗装やシールを準備して、自分好みのデザインにするに愛着もひとしおではないかと思いますが、今回はその余裕がなかったのでよくある透明のブックカバーシールで全体を覆ってみるに留めました。多少は汚れに強くなったと思われます。

## ▶ヤケドするようなハンダ付け▶

各種パーツのレイアウトも決まったので、あとは電気的な配線です。ハンダ付けは自分にあったペースで、それなりによいものを揃えた工具や材料を駆使しましょう。特にハンダ自体とコテには安物は厳禁です。できあがりの信頼性に大きく係わってくるからです。弘法筆を選ばずとはいいますが、よいものを選べるのに選ぶなという意味ではないのです。無理せず、慎重に手早く片付けていきましょう。くれぐれも、ハンダ付けの不良(イモハンダ)を、出さないよう

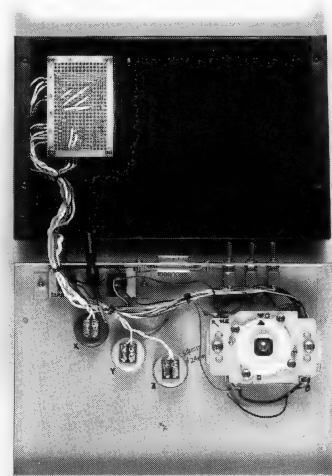


に注意してください。

制御基板の部品配置は穴から操作する関係上、スイッチとボリュームの位置を優先し、できるだけハンダ付け側の面で配線が交差しないことだけ気を配りました。これもケースのレイアウト同様に、行き当たりばったりで作業したので、目指していた効率や再現性には欠けるかもしれません。しかし、要は動けばよいわけで(信頼性を落とすようなことさえしなければ)、それほど問題にはなるような部分ではないでしょう。こだわるとキリがないので、蓋を閉めてしまえば見えなくなってしまうのをよいことに、これに関しては気にしないことにしました。

制御基板が完成したら、あとはスティックやレバーとの接続です。なぜかこだわっているSELECTとSTARTの配線図は図を参照してください。ここでダイオードは各レバーの信号の逆流防止の役目を果たしているので必ず入れるようにしてください。

初回にも書いたのですが、配線時に気をつけることは線が絡まないようにこまめに





まとめることです。そのためには、市販のケーブル縛りや、ビニール付きの針金などをうまく利用するとよいでしょう。粘着テープや輪ゴムで線を束ねた場合は、耐久性や安定性に不安が残るので、お勧めできません。

このほかにも、あとでわかるように配線材の色分けなどをしておくや修理やさらに改造するときに効果を発揮してくれます。備えあれば憂いなし、というところでしょうか。

また、配線にはできる限り余裕を持たせることも忘れてはいけません。蓋を開けてちょっと引っ張った程度で線が切れてしまうようでは、メンテナンスや修理の障害になります。あくまで邪魔にならない程度ながら、できる限りの余裕を確保するようにしてください。

## ▶完成したぞ、うれしいな◀

配線のチェックを終えたら、ついに完成です。実地のチェックも兼ねて、とりあえずいろいろゲームをプレイして幸せにひたってみました。

「悪魔城ドラキュラ」はSTARTボタンでポーズがかかる程度で、あまりどうということはありませんでしたが、編集部を持ち込んで「コットン」を試したところ、予想どおりの攻撃力が発揮されました。A+Bのボタンを設定し連射にして両方の連射シ

ョットとし、魔法の溜め射ち用にメインとボムのショットを通常ボタンとして2つ用意すると自由自在にコットンの攻撃を操ることができました。

ボタンの信号は連射よりも押しっぱなしのほうが優先されるため、両方押した場合魔法を溜めることになり、連射にしたショットボタンはほとんど押しっぱなしでOKでした(溜めに入る前に一瞬連射を離さないといけない場合がありますが……)。

この場合では、3ボタンであること、連射できること、ボタンの機能割り当てが自由なこと、そしておまけにスタートボタンでポーズがかかること、などの今回の目玉機能のほぼすべてを満喫できました。おかげで、X680x0版のコットンでは連射しながら火炎魔法で妖精を燃やして攻撃できることが発見できて(ちなみにオリジナルでは、連射していると燃えている妖精は敵を追尾しない)、ちょっとビックリもしました。

ほかにもいろいろ試しましたが、「スペースハリアー」では、まさに超連射とでも呼ぶべき高速連射が体験できるなど、いままでは気がつかなかったゲームの魅力が引き出せることもあるようです。コットン同様に連射だけでなく、溜め系の攻撃と使い分けるタイプの「出たな!! ツインビー」でも、やはりボタン機能の自由化がありがたく感じられました。

基本的に今回のジョイスティック改造で作成した機能は、特に自分にとっては目新

しいものではないのですが、今回、編集部でいろいろ試してみると、その機能が再確認できて非常に有意義だったと思います。なんか手前味噌のようですが、「究極」という目標は達成できたのではないかと思うのです。

## ▶次なる目標と野望◀

今回はジャンク同然のファミコン用ジョイスティックから始まりましたが、世の中にはまだ怪しげな悪魔の餌が転がっています。コナミから発売される、コマンド記憶の可能なスーパーファミコン用のパッドや、6つのボタンが並んだ各ゲーム機種用の例のゲーム専用ともいうべきジョイスティックやパッドなど、X680x0につながることできたら、役に立つかはさておいて、なにやら楽しそうです。

こういった怪しげな機器の対応はご要望と実力を天秤にかけて、不定期にやっぴいこうかな、などと思っています。原則的には対応していない便利そうな物体をX680x0につないでしまうといった方針で究極の向こう側を目指していきたいところです。どんどん怪しい方向に進んでいくようですが、それもまた定めなのでしょう。

とりあえず、ご要望や質問などは随時受け付けておりますので、またの機会を楽しみに待っててもらえると嬉しい限りです。それではまたお会いしましょう。

## ジョイスティックお手入れの手引き

はっきりいってしまえば、ジョイスティックというものは消耗品です。壊れかけたものをだましまし使っても不都合こそあれ、メリットはまずありません。しかしお気に入りの1台がすでに再度入手不可能であったり、独自の改造をしていた場合などは、故障したからといって買い替えて一件落着とはいきません。

こうした現実を考慮して、いくつかのジョイスティックのメンテナンスに関する注意事項や具体的なメンテナンスや修理の方法などにも触れておきたいと思います。

### ボタン

壊れるといちばん始末におえない部分で、なおかつ徐々に調子が悪くなっていくため、壊れたという見切りのつけにくいものです。基本的には全部交換して修理してしまうのがいちばんなのですが、ケース本体と一体化されているものは、まめに掃除して汚れを取るなどして、延命を図るほうがよいでしょう。スイッチ内部の接点がイカれることよりも、押す部分とかの傷

み具合や、ボタンの摩擦時に内部に溜まるプラスチックの粉に注意が必要です。

また通常、ボタンが効かなくなった場合というのは単に配線が外れていることが多く、深刻な故障である可能性は低いです。

### レバー

これも汚れるだけでなく、摩擦によって粉が溜まるものがあります。軸と玉はこまめに拭いて、軸の受けのプラスチックなどに少量の潤滑剤(CRCの556など)を吹き付けると、動きが滑らかになって遊びやすくなりますし、寿命も伸びると思います。

ただし、特定の方向が入り難いからといって、マイクロスイッチの板をいじるようなことはやめておいたほうがよいでしょう。逆に状態がひどくなる可能性が大了。

これもボタンと同様に、効かなくなったら配線が外れている心配をしたほうがよいでしょう。昨今の電子部品は個人でチャマチャ使っている程度で耐久限界を迎えるほどヤワではありません。

### 保証の問題

ここで、あっさり中を開けて修理することを勧めていますが、なかには開封すると保証が効かない旨の表示のあるジョイスティックがあると思います。しかし、保証といってもタダで直してくれるわけでもありませんし、事実目の前で壊れているものをなんとかしてはならない緊急性を考えれば、さっさと開けて、ハンダ付けで切れているところをつないだほうがよいに決まっています。人によって価値観は違いますが、たいして高くない(X680x0本体の値段と比べると、たぶんそうでしょう)ものですから、あまり小さなことにはこだわらないことをお勧めします。保証はしませんが……。

あと、最後にジョイスティックのメンテナンスとしていちばん大事なことは、ゲームをする前に手を洗うことです。このように日頃から気を配ってこそ、ジョイスティックも調子よくあなたの遊びのパートナーであり続けられるのです。

FISH.Xに続け!

# スクリーンセーバーのモジュールを作る

Ishigami Tatsuya 石上 達也

石上版スクリーンセーバー用のモジュールの作り方を解説しましょう。

ひたすら艶やかに画面を彩るもよし、実用的に画面を消すもよし。

SX-WINDOWでの作業が楽しくなるようなモジュールを制作してみてください。

皆さん先月号のプログラムは実行したでしょうか? え、なんのプログラムかって? やだなー、スクリーンセーバーに決まっているじゃないですか。

とにかく、あのストライダー横内氏が1カ月間かかりきりで作成したアニメーションです。感動しないわけがありません。

そんなことはありませんと思いますが、読者の方で、万が一にも、なにかの間違いで、熱帯の海をまだ満喫されていない方がいましたら、ただちに先月号を読み返してください。

ね、ね、感動したでしょう。

で、感動したら、次はその作品を超えるようなものを自分で作ろうとするのが人情というもの。先月号の操作編に続き、今月はスクリーンセーバーモジュールのプログラミング編です。

## シェルとモジュールの関係

付録ディスクをそのまま解凍すると、SAVER.Xというファイルネームになっていました。これではウィンドウの名前が「画面暗前」なのにファイル名がSAVER.X、という紛らわしい状態になってしまいますので、ファイル名を「画面暗前.X」にリネームしておいてください(トホホ)。

さて、付録ディスクと一緒に入っていたFISH.XとかQUIX.Xというファイルを画面暗前のウィンドウに放り込むと、ウィンドウ内の表示がいろいろと変わります。FISH.XやQUIX.Xは自分のウィンドウを開くことはしません。その代わり「画面暗前」のウィンドウ内に、いろいろなコントロールを出現させたり、消去させたりします。

正確にいうと、他人のウィンドウですから、(画面暗前に)出現させてもらったり、消去してもらったりします。

唐突ですが、「画面暗前」をアパートの大

家さんにたとえると、QUIX.XとかFISH.Xとかは、そのアパートの住民ということになります。部屋にエアコンをつけたいので、壁に穴を開けてもいいですか。はい、どうぞ。ホットカーペットを買ったんで、ブレーカーを変えてもいいですか。はいはい、どうぞってなもんです。

このような関係が成立するとき、大家さんを「シェル」、住民を「モジュール」と呼びます。

なぜ、大家さんがシェルで、住民がモジュールなのかというと、図1です。

モジュール(Module)というのは大ざっぱにいうと、「部品」という意味です。構造化プログラミングの話によく出てくる単語ですね。交換可能な部品としての住人、というなんかのレトリックだとしたら「?」な表現ですが、とにかく、この場合、住人はモジュールです。

シェル(Shell)というのは、貝の殻の部分のことです。アパートの大家さんと、貝殻が結びつきづらい人は、貝の具の部分を住民と考えてください(ヤドカリでも可)。

スクリーンセーバーのプログラムには、キーボードやマウスを監視して、タイマーを見計らって、その見計らう時刻を設定して……などという部分が必ず含まれます。どのように暗転するか、というところでスクリーンセーバーの個性は競われるべきなのに、そのような雑用をクリアしないと競うことができないのです。

これはいけません。

往々にして、このような場合には、共通のサブルーチンをライブラリ化するものなのですが、今回はしていません。この方法では、ひとつモジュールを作るたびに、同一のライブラリがリンクされてしまいます。30個のモジュールがあれば、30個の同じライブラリが、メモリ上なり、ディスク上なりに存在することになるのです。また、ライブラリにバグが見つかるたびに、あるい

はライブラリがバージョンアップをするたびに、リンクをやり直す必要が出てきてしまいます。

もともと1種類しかないはずの共通サブルーチンがあっちこちに存在するから、メモリを余計に消費したり、バージョン管理が煩雑になったりするのです。1種類のサブルーチンパッケージを、ずっと1カ所にまとめておけばこのようなことは起こりません。アパートの大家さんが部屋にコンロとか流しとかを用意してくれれば、新しい住人はそれらを買って揃える必要もなくなるし、出ていく人は、それらを持って出ていく必要もなくなるのです。

共通部分を1カ所にまとめておく、ということは、モジュールごとに組み込まない、ということです。従来ならひとつのプログラムに組み込まれていたものを、カスタム部分(モジュール)と共通部分(シェル)に分断してやります。そして、2つ目のプログラムからはカスタム部分のみをメモリ上に展開し、共通部分はひとつ目のプログラムと共有するのです。ここで、えっ? と思うかもしれませんが、SX-WINDOWはマルチタスクシステムなのです。複数のプログラムをいっぺんに走らせることもできれば、走っているプログラム同士の関係も設定できるのです。

コイツは変なことばかりいって読者に混乱を引き起こそうとしているな、と思った方はMS-WINDOWSの参考書をチラッと読んでください。きっとDLL(Dynamic Linking Library)という名前があると思います(ということは、よく知らないけどMacintoshにもあるんだろうな、きっと)。

C言語では、プログラム中でライブラリや下位のサブルーチン(関数)を呼び出すときにはスタックにパラメータを積んでいきます。呼び出されたサブルーチンはその積み上げられた値をパラメータとして参照します。ひるがえって、SX-WINDOWで



は、他タスクにある前述のようなサブルーチン呼び出す方法というのは決まっています。ま、私がいきなり使い始めたことですので、誰かが決めてくれるわけではないんですけどね。

## メッセージのフォーマット

そんなこんなで、図1のような関係にあるシェルとモジュールは、5月号で説明したような「タスク間通信」と呼ばれる方法を使ってメッセージのやり取りを行うことにしました。細かい説明は5月号で行いましたので、適当に参考にしてください。

詳しくは後述しますが、MODULE.Hというヘッダファイルにもメッセージを発信する関数がsendMesという名前で収めてあります。この関数は、

```
sendMes(int id, char *文字列)
```

と使うことによって、idで示されるタスクIDを持つタスクへ文字列を送ることができます。

現在のバージョンではモジュールからシェルへ送ることのできる文字列に以下のものが用意されています。

### ○NONE n

コントロール類を消去します。

シェルには、4つのコントロールを配置することができます。このコントロールには、上から1,2,3,4と番号がついていて、この番号により識別されます。ここではその番号のことをnと表しました（以下同じ）。

たとえば、

```
sendMes(taskBuf->parentid,  
"NONE 1");
```

という関数を実行すると、シェルの1番目の位置に描かれていたコントロールを消去します。コントロールハンドルなどの破棄も自動的にシェルが行うので、細かいことは気にしなくて結構です。

### ○SN n, 文字列, 数値

n番目の位置に、ボリューム (Slider with Number) を配置します。

このとき、文字列が指定されていれば、その文字列をキャプションとしてボリュームの上に描画します。

数値が指定されていれば、その値をボリュームの初期値とします。

このボリュームの取り得る値は0~100までの整数値です。

### ○PB n, 文字列1, 文字列2

n番目の位置に標準ボタン (Push Button) を配置します。

文字列1が指定されていれば、その文字

列をキャプションとして標準ボタンの上に描画します。

文字列2が指定されていれば、その文字列を標準ボタンの中に書き込みます。

### ○CB n, 文字列1, 文字列2, 数値

n番目の位置に、オルタネイトボタンを描画します。名前がCBなのは、私が最近までオルタネイトボタンのことをチェックボタンだと勘違いしていたからで、他意はありません。四角の中にチェック模様を入れるのだから、こっちのほうがしっくりくると思うんだけどなあ。

それはともかく、例によって文字列1が指定されていればオルタネイトボタンの上にキャプションとして描画します。

文字列2が指定されていれば、オルタネイトボタンの右横にそれを描画します。

数値が指定されていれば、その値をオルタネイトボタンの初期値とします。この値は0で四角の中にチェックマークなし、0以外でチェックマークを描画します。

### ○UB n, 文字列1, 数値, 要素文字列群

n番目の位置に、アップダウンボタン (Updown Button) を配置します。

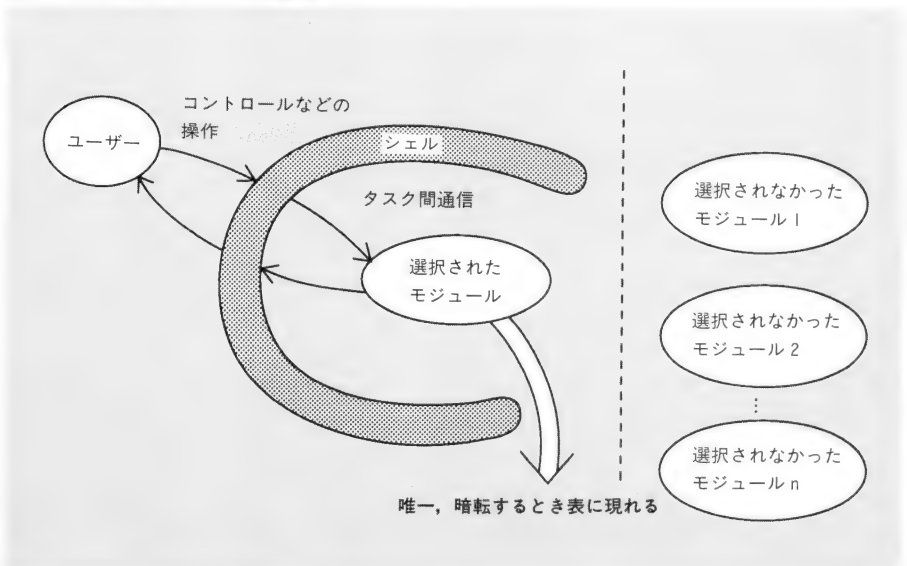
文字列1が与えられていれば、アップダウンボタンの上にキャプションとして描画します。

数値が与えられていれば、その値をアップダウンボタンの初期値として代入します。

アップダウンボタンの要素文字列群は、「,」で区切られた文字列の集合です。アップダウンボタンの値がXのときには、アップダウンボタンのテキスト領域にX番目の要素文字列が描画されます。

アップダウンボタンは、1から要素文字列の数までの範囲の値を取ります。

図1 シェルとモジュールの関係



たとえば、

```
sendMes(taskBuf->parentid,  
"UB 1, アップダウンボタン,"  
"2, 始め, 次ぎ, 終わり");
```

という命令によって配置されたアップダウンボタンは初期状態で、「次ぎ」というテキストを表示します。アップボタンを押すことによってアップダウンボタンの持つ値は初期値の2から3へと変化し、「終わり」というテキストが表示されます。この場合、アップダウンボタンが取り得る最大値は3なので、これ以上アップボタンを押してもなにも変化しません。

この状態で、ダウンボタンを押すとアップダウンボタンの持つ値は3から2へと変化し、テキストは「次ぎ」と元へ戻ります。さらにダウンボタンを押すとアップダウンボタンの持つ値は1になり、テキストは「始め」となります。この1という値はアップダウンボタンの取り得る値の下限ですから、これ以上ダウンボタンを押しても、なにも変化は起こりません。

### ○COM1 文字列

「画面暗前」のシェルを見てください。下のほうに、白く塗られた四角形があります。この四角形には、半角で22文字までの文字列を2段描画させることができます。この命令は、その四角形の上段に、指定された文字列を描画させます。

### ○COM2 文字列

COM1とほとんど同じですが、文字列を描画する場所が四角形の下段になります。

たとえば、

```
sendMes(taskBuf->parentid,  
"COM1 BLANK.X (C) T.Ishigami");
```



写真1 BLANK.Xをドラッグしたとき

```
sendMes(taskBuf->parentid,
"COM2 画面のコントラストを下げま
す");
sendMes(taskBuf->parentid,
"SN 1, ボリューム 1, 30");
sendMes(taskBuf->parentid,
"PB 2, 押しボタン 2, 押せ");
sendMes(taskBuf->parentid,
```

"CB 3, チェックボックス 3,"  
 "←これだよ, 1");  
 sendMes(taskBuf->parentid,  
 "UB 4, アップダウンボタン 4,"  
 "2, 始め, 次ぎ, 終わり");

というプログラムを実行した場合、シェルは写真1のような状態になります（シェルにBLANK.Xを放り込んだときの状態です）。

## プログラムの実際

シェルとモジュールの関係、通信内容がわかったところで、モジュールをプログラミングする方法を説明します。

シェルとモジュールの関係はモジュールの種類によらず一定です。シェルが暗転/復帰のタイミングを作成し、タスク間通信に

よって、モジュールに教える。モジュールは、そのメッセージのとおり動作する。ここらへんは前述のとおりです。

このシェルからモジュールに送られてくるメッセージは数種類しかありませんから、メッセージを受け取り必要なサブルーチン/関数を呼び出す部分はどのモジュールでもだいたい同じになるはずです。

モジュールを作成するときに同じような関数を書き直すのは大変ですから、使い回しのできるようなかたちにして作っておきました。ですから、プログラマはシェル→モジュールという方向に渡されるメッセージについて直接関与する必要はありません。それらの処理を行う関数はMODULE.Hというファイルに入っています。

本来なら、このような関数はライブラリのかたちにしてモジュールにリンクさせる

## リスト MODULE.H

```
1: /*
2:  画面暗前モジュールヘッダファイル
3:
4:  Programmed By T.Ishigami
5:  9/25/92
6: */
7:
8:
9: #define      SX_BASIC_SEND      258
10:
11: #define      FALSE      0
12: #define      TRUE      FALSE
13:
14: #define      EVENTMASK      EM_EVERY
15:
16: event      eventRec;
17:
18: main()
19: {
20:   if ( SX_init() == FALSE ) {
21:     DMErrror( 0x101, "ウィンドウがオープンできません" );
22:     SX_term();
23:   }
24:   while( 1 ) {
25:     TSEventAvail(EVENTMASK, (tsevent*)&eventRec);
26:     switch( eventRec.what ) {
27:       case E_IDLE:      drawFrame();      break;
28:       case E_UPDATE:    procUPDATE();      break;
29:       case E_SYSTEM1:   procSYSTEM();      break;
30:       case E_SYSTEM2:   procSYSTEM();      break;
31:     }
32:   }
33: }
34:
35: SX_init()
36: {
37:   task      taskBuf;
38:
39:   TSGetTdb(&taskBuf, -1);
40:
41:   /* -AD オプションがついていなかったら起動しない */
42:   if( strcmp(taskBuf.command.Lstr, "-AD") ) exit(-1);
43:
44:   taskBuf.command.length = 0;
45:   taskBuf.command.Lstr[0] = 0;      /* -AD オプションを消す */
46:   TSSetTdb( &taskBuf, -1);
47:
48:   sendInitialMes(&taskBuf);
49:   initialize(&taskBuf);
50:   return( TRUE );
51: }
52:
53: SX_term()
54: {
55:   doRecov();
56:   exit(1);
57: }
58:
59: procSYSTEM()
60: {
61:   char      **mess;
62:   char      com[20]={0}, arg[90]={0};
```

```
63:
64:   switch( ((tsevent*)&eventRec)->what2 ) {
65:     case CLOSEALL:
66:     case ENDTSK:
67:       SX_term();
68:       break;
69:     case SX_BASIC_SEND:
70:       mess=(char**)((tsevent*)&eventRec)->whom;
71:       sscanf( *mess, "%s %s", com, arg);
72:       if ( strcmpi( com, "DARK" ) == 0 ) {
73:         doDark();
74:       } else if ( strcmpi( com, "DEMO" ) == 0 ) {
75:         doDemo();
76:       } else if ( strcmpi( com, "RECOV" ) == 0 ) {
77:         doRecov();
78:       } else if ( strcmpi( com, "CLOSE" ) == 0 ) {
79:         SX_term();
80:       } else if ( strcmpi( com, "CONTROL" ) == 0 ) {
81:         changeControl(arg);
82:       } else if ( strcmpi( com, "DIALOG" ) == 0 ) {
83:         doDialog();
84:       }
85:       break;
86:   }
87: }
88:
89: /******
90:   文字列を送信する
91:   *****/
92: sendMes(int id, char *str)
93: {
94:   95:   char      **hdl;
96:   96:   int      ret;
97:   97:   int      cnt = 10; /*メッセージのやりとりを10回行って
98:                       それでもダメならエラーにする */
99:   99:   tsevent      eventRec;
100:
101:   if(id < 0) return(FALSE);
102:
103:   hdl = (char **)MMChHdlNew(strlen(str));
104:   if(hdl == NULL) {
105:     DMErrror(1, "メモリが確保出来ません");
106:     return(FALSE);
107:   }
108:   strcpy(hdl, str);
109:
110:   eventRec.whom = (long)hdl;
111:   eventRec.when = EMSysTime();
112:   eventRec.what2 = SX_BASIC_SEND;
113:
114:   do {
115:     ret = TSSendMes(id, &eventRec);
116:     cnt--;
117:   } while(ret != 0 && ret != 2 && cnt != 0);
118:
119:   MMHdlDispose(hdl);
120:   if(cnt == 0) {
121:     DMErrror(1, "無効なタスクに通信を行ないました");
122:   }
123:   return(TRUE);
124: }
```



べきかもしれません。が、どのみちヘッダファイルが必要となることすし、複数のファイルがあったりすると、かえって煩わしくなるので諸定数の定義と共に必要な関数類もヘッダファイルに入れておきました。

このヘッダファイルの中には、SXプログラムでいうところのスケルトン部分も含まれていますので、ユーザーは、そのスケルトンから呼び出される関数を書くだけで、モジュールが作成できます。

ユーザーが作成する関数は、以下のとおりです。

#### ○initialize(task \*taskBuf)

モジュールの初期化を行います。引数taskBufには、モジュールのタスクバッファポインタが入っています。

#### ○sendInitialMes(task \*taskBuf)

シェルのコントロール類を描画させます。シェルのタスクIDはグローバル変数taskBufに収められていますので、そのタスクへ前述のメッセージを送信すれば、コントロール類の描画を行うことができます。

#### ○changeControl(char \*arg)

シェル上に描画されたコントロール類が操作されたときに呼び出されます。引数argには、

操作されたコントロールの番号,新しい値

という形式で文字列が入っていますので、  
sscanf(arg, "%d,%d", &cnt, &var);  
で、int型変数cntに変更されたコントロールの番号、int型変数varに新しい値を取り込むことができます。

もし必要ならば、この関数内でコントロールの値の変更に対応する動作を行ってください。

#### ○procUPDATE(void)

必要場合は画面のアップデート処理を行ってください。

この関数が呼び出されるのは、デモモードで、ほかのウィンドウが移動された場合です。デモモードでない暗転（つまり、キーボードやマウスが一定時間操作されなかったときに起こる暗転。なにか用語を造ったときによかったな）中のときは、ほかのウィンドウが操作されたということはマウスが操作されたということですから、アップデート処理ではなく暗転の終了処理を行います（後述のdoRecovが代わりに呼ばれる）。

#### ○doDark(void)

暗転を開始する。この場合の暗転はデモモードではありません。デモモードでない暗転（同上）です。

#### ○doDemo(void)

デモモードで暗転を開始する。モジュールから見てデモモードとそうでない暗転の違いは開始時に呼ばれる関数がdoDarkかdoDemoかの違いしかありません。必要ならば、フラグを設けて呼ばれた関数によってモードの設定を行ってください。

#### ○drawFrame()

アイドルイベントに対応する関数です。暗転中にアニメーションパターンなどを用いている場合には、この関数内で書き換えを行ってください。

この関数は、暗転中である/ないにかかわらず呼び出されますので、必要な場合にはユーザーがその判別を行うようにしてください。

#### ○doRecov(void)

暗転中の画面を復帰します。

#### ○doDialog()

シェル上のDIALOGボタンが押されると呼び出されます。この関数内でなにを行わせるかは、ユーザーの自由ですが、先月の付録ディスクに収録されていたプログラムをとりあえずの規範とします。

### 画面の消去について

画面を暗転させるには、ほかのウィンドウを消して、画面を真っ暗にしなければなりません。画面を真っ暗にすればそれでいいかというと、暗転を終了させるときに、表示内容の復帰もできなくてははいけません。

その方法として真っ先に思い浮かぶのが、真っ暗なウィンドウを開く、というやつです。これならば、ウィンドウを閉じるだけで、復帰作業が行えますから便利です。QUIX.Xなどがこの方法を用いています。

しかし、この方法では、どうしてもシステムアイコンやデスクアイコンを消去することができません。しょうがないので、FISH.XではCRTICを直接叩いて、テキスト画面の表示を行わないようにして、この問題を回避しています。

また、いきなり画面を真っ暗にしないでデスクトップ画面に手を加えていき、徐々に画面を暗転させていきたいという場合があるかもしれません（例：MELT.X）。

このような場合は、ウィンドウがどこに開かれていようがお構いなく画面を書き換えていくわけですから、グラフポートをセットして、ウィンドウマンを呼び出して……というような手段は使えません。画面の描画は、IOCSを呼び出すことになります（例外としてPUZZLE.Xは、処理速度の問題

で直接G-RAMを操作しています）。このような描画は、SX-WINDOWの関与しないところで行われますから、暗転から画面を復帰する場合もSX-WINDOWの機能は使えません。

その対応策として、真っ先に思い浮かぶのが、

SX-WINDOWの関与しない手段で画面を保存する

ということになります。適当なメモリエリアにG-RAMからデータを転送する、というのが一般的でしょう。

画面情報を保存したいから、メモリに転送して保存する

はい、非常に正直な手段です。

しかし、SX-WINDOWの画面は、768×512 ドットの4プレーン構成ですから、ちょっとした計算を経て、約197Kバイトのメモリ領域が必要となってくるわけです。隠れ機能を使ってインタレースをかければ、さらにその倍のメモリ容量が必要になってきます。

これが、MS-WINDOWSだったら、「てやんでえ」とかいいながらも許せる値なのですが、SX-WINDOWだとなんとなくもったいない大きさです。6月号によるとX68000ユーザーの平均搭載メモリは3Mバイトですからね。これがワープロとかドロソフトだったら許せるのですが、スクリーンセーバーなんて、しょせんはアクセサリですからね。

で、画面の復帰とはいいいながら、もう一度画面を描き直そうという方法にたどり着きます。この場合の描き直すというのは、どっかから、メモリデータを転送してくるのではなく、ウィンドウにいろいろ描画するプログラムをもう一度呼び出して、ウィンドウを描き直してもらおう、ということです。

ウィンドウを描き直してくれ—というお願いは、そのウィンドウへアップデートイベントを送りつける、ということです。えーと、確か、タスクマンに任意のタスクに指定したイベントを送れるコールがあったな。実行されているタスクのIDも調べがつかない。えー、でもそれをプログラムするの？ 面倒臭いなあ。と、思うかもしれませんが、あと一歩です。

なにも、アップデートイベントを強引に送りつけてやらなくても自然とアップデートイベントが発生するような環境を作ればよいのです。

さて、問題。自然とアップデートウィンドウが発生する状態とは？

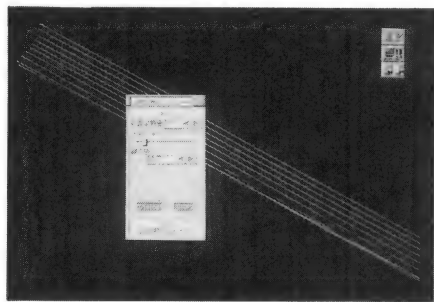


写真2 デモモードのとき (QUIX.X)

答) ウィンドウの移動/消去が起こったときです。

というわけで、暗転に用いた領域に、ウィンドウを広げて、消去してやればよいのです。いままで、ウィンドウの復帰ばかりを考えていましたが、実際には背景をいじった場合には、それも戻してやらなければいけないわけで、この「ウィンドウを開いて、すぐに閉じる」、という方法はその点でも有効です。

ただし、暗転を始めるときにウィンドウを広げてはいけません。それでは、デスクトップ画面が隠れてしまいます。いまは、デスクトップ画面をいじったあとに、どのようにして復帰させるかということを考えていたのです。

また、この方法もただ普通にウィンドウを開いて閉じればよいというものではありません。ウィンドウをなにも考えずにオープンすると、灰色のウィンドウを開いてしまいます。つまり、暗転を解除→画面が灰色に塗り潰される→デスクトップ画面が復帰、という構成になります。最終的にデスクトップ画面を復帰させたいだけなのに、2番目の作業は明らかに無駄なステップです。

結論から先にいってしまえば、解決方法は以下ようになります。

- 1) 暗転の際、不可視のウィンドウを開いておく (不可視なのでデスクトップ画面は保存される)
- 2) 暗転から復帰する際、1)のウィンドウを可視化する
- 3) ウィンドウをもっとも手前に持ってくる
- 4) ウィンドウを消す

以上のような方法を使えば、先ほどのような問題を解決することができます。

ただし、この方法にもひとつだけ問題点があります。SX-WINDOWの公開されている機能だけでは、どんなに頑張ってもシステムアイコンより手前にウィンドウを開くことができません。暗転させる領域にシステムアイコンやデスクアクセサリアイコ

ンがあった場合、見事に破壊されます。あ、破壊といっても別にプログラムが暴走するわけではありません。アイコンのグラフィックが破壊されるだけです。この状態が嫌ならOPT1キーを押しながら、マウスで左クリックしてやるなどの方法により強引にアップデートすると、元の状態に戻ります。

5月号でも述べましたが、シャープから発売のデスクアクセサリ集に収録されているスクリーンセーバーはなぜか、これやってくる。シェルから(?)モジュールに渡される構造体の中に、graph \*portという変数があって、これをグラフポートにセットして塗り潰せばシステムアイコンは綺麗に消えますし、破棄すればシステムアイコンは復活する、という具合です。

アプリケーションが強引にSX-WINDOWを拡張しているとは考えづらいので、探せばこのようなグラフポートを取得できるようなSXコールが隠れているはず。強引に探し出せないこともないですが、隠れているということは探してくれるな、ということでもありますから、SX開発キットの発売を待って対応を考えようと思います。

## デモモードについて

画面暗前には、すぐさま暗転を開始させるデモモードという機能がついています。

たとえば、新しいモジュールをすぐに試してみたいときや、モジュールのデバッグを行おうというとき、X68000の前に座って、じっと暗転が開始されるのを待ち続けるのはけっこうマヌケです (最低でも、1分間)。このように、本来のスクリーンセーバーとしての用途以外で、モジュールの実行をすぐさま開始させたいときに使う機能です。

例として、付録ディスク中のFLYING.Xを実行してみてください。実際に暗転するときには、画面全部を使ってヤカンが飛ぶかウアニメーションが表示されますが、デモモードで暗転したときには、画面暗前のシェルの画面が残ったままです。また、デモモードからの復帰は、シェルのSTOPボタンが押されたときのみ行われますので、デモ中はいくらマウスをいじっても画面の復帰は行われません。このことを利用して、QUIX.XやFLYING.Xのように暗転中に参照されるパラメータを調整する機能を実現することができます。

で、5月号で、SX-WINDOWの機能では、どーたらこーたらいろいろ書きましたが、うまい解決方法が (ディスクの縮め

切り直前に)、思い浮かびましたので、紹介します。

SX-WINDOWの公開されている機能では、「上から2番目の位置にウィンドウを新たに開く」ということはできません。SX本を隅から隅まで探せば、近いことは書いてあるかもしれませんが、私は見つけることができませんでした。

1回の行為で実現できないので、以下のような複数回の手順を踏んでやります。

- 1) しょうがないから、暗転用のウィンドウをいちばん手前に開いてしまう (シェルのウィンドウは隠れてしまう)。ただし、適当なフラグに、本当はシェルの画面をいちばん前に出したいんだ、ということメモしておく。
- 2) アイドルイベントが初めて回ってきたときには、先ほどのフラグの内容を調べる。もし、シェルのウィンドウをいちばん手前に出すべきだったら、シェルのタスクに、「ACTIVE」という命令を送る (5月号を参照)。

これで、シェルのウィンドウが一瞬チラついてしまうものの、目的とする機能を実現することができます。

## 最後に

付録ディスクが読者のもとへ届いて約1カ月。掲載されていたプログラムを初めて実行したときの感動もそろそろ薄れてきた頃かもしれません (自分で書いてて恥ずかしい言葉だな。以下続く)。あるいは、もっと凄いモジュールのアイデアが、はっきりとしたイメージになってきた頃かもしれません。

この記事を読んで、自分もモジュールを作ろう、と思ってくれる読者が現れたら、素晴らしいと思います。

そのようにしてできた作品を編集部ではお待ちしております。また、制作の過程で問題点が出てきた場合には質問箱へご連絡くだされば、できる限りのサポートをするつもりです。

そういえば、SXMookと並行して (ひょっとすると、SXMookの一部になるかもしれないし、先に出てしまうかもしれないし、結局詳しいことはなにも決まっていない)、スクリーンセーバーのモジュール集を出すとか出さないとかいう話がありましたから、そちらのほうに採用されるかもしれません。

アクセラレータは必ず動かしますから待ってください。んじゃ。



# AD PCMを使ってメディアコンバート CASSAVE.X / CASLOAD.X

Harashino Makoto 原篠 誠

必要なものはオーディオケーブル1本だけ。AD PCMを使ったデータコンバートプログラムです。実用度は?ですが、フォーマットさえわかれば、カセットベースでのデータコンバートもできるユニークなツールです。

Compactシリーズの出現以来、X68000の世界にも3.5インチのフロッピーディスクが普及しつつあります。このことは満開製作所より突如として現れた、REDZONEにより一層拍車がかかることでしょう。そこで問題となるのがメディアコンバートです。

5インチと3.5インチ。異なるディスクで同じデータを共有するにはどうすればよいでしょう。まず考えられるのが増設ドライブ。それからRS-232Cを使う方法。残念なことにどちらもいくらかの出費が必要です。次にダンプリストを自力で打ち込む。根気のある人はそれもいでしょう。

ほかになにか方法はないでしょうか。Oh!Xの1993年7月号にケーブルを自作する方法が載っていますが、ほんの少しハードの知識が必要なうえにどうやらあまり安全ではないようです。またジョイスティック端子を使う方法とか、MIDIボードを使う方法などいくつか考えられますが、いずれもハードを自作しなければなりません。

「ハードなしでメディアコンバートなんてそんなうまい話ないよ」という声が聞こえてきそうですが、まだ方法は残されていたのです。古くはパピコンの頃から使われていた方法、そう、音声データを使うのです(笑)。都合のいいことにX68000には音声入力端子があります。うまいことPCMデータに加工してやればできないことはないでしょう。

## 動作原理

いまをさかのぼること十数年。当時フロッピーディスクなど高嶺の花で、データのセーブといえども知る人も少ないカセットテープを使っていたものでした。このデータ転送方式は現在でもパソコン通信などに使われています。

その方法とはデータを1ビットごとに特定の周波数のサイン波に乗せていくのです。

この周波数のことをボーレートといいます。具体的にはビットが0の場合は1周期、1の場合は2倍の周波数で2周期の間サイン波を出力すればよいのです。ということは0と1どちらの場合も、出力する時間の長さは同じなので1秒間にボーレートと同じだけのビットデータを、出力することになります。

しかし、すべてのデータを1ビットずつ垂れ流していけばよいというわけではありません。実際には1バイト(=8ビット)ごとに頭に0を1ビット、最後に1を2ビットつけて合計11ビットにして出力するのです。この頭の1ビットをスタートビット、後ろの2ビットをストップビットと呼びます。例を挙げると、58<sub>H</sub>なら2進数で01011000<sub>B</sub>となり、スタートビット、ストップビットをつけると11010110000<sub>B</sub>となります。ボーレートが1200ボーならば1秒間に1200ビット、ということは、

$$1200 \div 11 \approx 109 \text{ バイト}$$

のデータを転送できることになります。

ほかにヘッダというものがあります。実際にカセットテープを使っていた人にはわかると思いますが、音声データの最初にくピーという部分のことです。これは1が数千ビットの間出力されているものでデータとしての意味はありません(同期を取るため?)。1ということは2倍の周波数なのでボーレートが1200ボーなら2400Hz(オクターブ6のD#くらい)になります。心当たりがあるでしょう。

## プログラムの説明

CASSAVE.Xは、バイナリファイルを16ビットPCMデータに変換するものです。よってPCM→AD PCM変換ができるプログラム(ZVT.Xなど)が必要になります。“-F”スイッチでサンプリング周波数、“-B”スイッチでボーレートを変更できますが、

ボーレートはサンプリング周波数の4分の1以下でないとうまくPCMデータ化できません。15600Hzでサンプリングするなら3600ボー程度。逆に1200ボーなら7800Hzでデータ化すればよいということになります。

CASLOAD.Xは16ビットPCMデータをバイナリファイルに復元します。最初は微分すればなんとかなると思っていましたが、ノイズに弱い、1サイクルの切り出しがうまくいかないなどの理由によりいき詰まってしまいました。そこで思いついたのが逆フーリエ変換する方法です(石上達也さんに感謝)。詳しいことはOh!Xの1991年12月号を参照してください。

## 使用例

それでは実際にどうやって転送するのか手順を追ってみましょう。仮に転送したいファイルをTEST.DOCとします。まず、PCMデータに変換します。

```
CASSAVE TEST.DOC TEST.P16 -f4 -b2400 -a1000 -h0
```

これでTEST.P16というファイルができました。ボーレートはもう少し上げてもかまいません。“-a1000”というのは振幅のことで、まあ音量みたいなものです。

今度はX68000で使えるAD PCM方式に変換します。ZVT.Xを使うなら、

```
ZVT -A TEST.P16 TEST.PCM
```

とすればよいでしょう。そしてできあがったファイルを、

```
COPY TEST.PCM PCM
```

などとして、再生します。その音データを直接でも間接でもよいですから、受け側のX68000に送り込みます。録音にはZVT.XのTRIGGERモードを使うとよいでしょう。そうして録音したファイルをTEST2.PCMとすると、今度はZVT.Xで、

```
ZVT -C TEST2.PCM TEST2.P16
```

として16ビットPCMに変換してから、

CASLOAD TEST2.P16 TEST2.DO  
C -f4 -b2400 -m100  
とすればやっと転送が終了したことになります。“-m100”は振幅が100以下の部分をノイズとみなして無視するという意味です。うまくいったら元の名前にリネームしておきましょう。

## 応用例

せっかくカセットフォーマットに変換できるのでから、MSXにデータを落とす方法を紹介します(本当はS-OSに落としたりはできなかったけど資料がないので……)。

まず、MSXのカセットテープのデータフォーマットを知らなければなりません。BASICプログラムは、中間コードやリンクポインタなど説明が長くなるので、ここではバイナリデータについて説明しましょう(BSAVEやBLOADのことです)。

最初に、8000ビットのヘッダのあとに識別コードが10バイト、ファイル名が6バイトと続きます。そしてしばらく無音状態になって、2000ビットのヘッダ、スタートアドレス、エンドアドレス、実行アドレスと続き、最後にバイナリデータがきます。識別コードとはフォーマットを知らせるもの

で、ここではD0<sub>H</sub>です。

ここではファイル名とバイナリデータの2つに分けてPCMデータ化します。まずはファイル名から。これはたったの16バイトなのでデバッグやMACINTOSH-Cを使えば簡単にできるでしょう。次にバイナリデータのほうですが、最初の3つのアドレスは2バイトのインテル並び(上位バイトと下位バイトが逆)で指定します。また、データの長さは(エンドアドレス)-(スタートアドレス)+1になります。E000<sub>H</sub>から400<sub>H</sub>バイトロードしたいなら、

00,E0,FF,E3,00,E0……

となります。こうやって作った、2つのファイルをそれぞれFILENAME.BIN、BIN DATA.BINとすると、

```
/* 16ビットPCM形式に変換
CASSAVE FILENAME.BIN
FILENAME.P16 -f2 -b1200 -h1000
CASSAVE BINDATA.BIN BIN
DATA.P16 -f2 -b1200 -h1000
/* AD PCM形式に変換
ZVT -a FILENAME.P16 FILENAME.PCM
ZVT -a BINDATA.P16 BINDATA.PCM
```

としてから2つのPCMファイルを続けて

## リスト1 CASLOAD.C

```
1: /*
2: * CASLOAD.C version 1.00 Copyright 1993 M.Harashino
3: */
4:
5: #include <stdio.h>
6: #include <stdlib.h>
7: #include <math.h>
8: #include <ctype.h>
9:
10: #define stblsize 256
11: #define stblmask 0x00ff
12: #define stblamp 4096
13: #define pi2 stblsize/4
14:
15: unsigned short freqtable[]={3900,5200,7800,10400,15600};
16:
17: unsigned int freq,rate,minlevel,counter,info;
18: short sintable[stblsize];
19: char *fn1,*fn2;
20: FILE *fp1,*fp2;
21:
22: void help()
23: {
24:     puts( "使い方: CASLOAD <file name> [<file name2>] [<swi
25: tch>]");
26:     puts( "  -f<num>: サンプル周波数設定 (default: 1560
27: 0) <num>=0: 3900");
28:     puts( "  -b<num>: ボーレート設定 (default: 120
29: 0) 1: 5200");
30:     puts( "  -m<num>: ノイズの最大振幅 (default: 10
31: 0) 2: 7800");
32:     puts( "  -i : 画面にもアスキーコードで出力する
33: 3: 10400");
34:     puts( "  -h or ?: ヘルプメッセージ
35: 4: 15600");
36: }
37:
38: void cmdget(int argc, char *argv[])
39: {
40:     int i;
41:     char c;
42:
43:     freq=4;
44:     rate=1200;
45:     minlevel=100;
46:     info=0;
47:
48:     fn1=fn2=0;
49:
50:     for (i=1; i<argc; i++) {
51:         c=argv[i][0];
52:         if (c == '-' || c == '/') {
53:             switch(tolower(argv[i][1])) {
54:                 case 'f':
55:                     freq=atoi(argv[i][2]);
56:                     break;
57:                 case 'b':
58:                     rate=atoi(argv[i][2]);
59:                     break;
60:             }
61:         }
62:     }
63:
64:     if (freq > 4 || rate > freqtable[freq]) {
65:         puts("パラメータの指定に間違いがあります");
66:         exit(-1);
67:     }
68:
69:     freq=freqtable[freq];
70:
71:     if (fn2 == 0) {
72:         help();
73:         exit(-1);
74:     }
75:
76:     void stblmake()
77:     {
78:         int i;
79:
80:         for (i=0; i<stblsize; i++) {
81:             sintable[i]=sin(pi*2*i/stblsize)*stblamp;
82:         }
83:     }
84:
85:     void movetop()
86:     {
87:         short data;
88:
89:         while(feof(fp1) == 0) if ((data=ungetc(getc(fp1))) > min
90: level) break;
91:         fseek(fp1,-2,1);
92:     }
93:
94:     int pow2s(int num)
95:     {
96:         num/=stblamp;
97:         return(num*num);
98:     }
99: }
```

再生してカセットテープに録音すればよいでしょう(再生周波数に注意)。ヘッダは申しわけ程度につければ十分です。

今度はMSX側で、

BLOAD"CAS:<ファイル名>"

とやってロードします。うまくいかない場合は、CASSAVE時に“-P”スイッチをつけてもう一度やり直してみましょう。

## 最後に

最初は我が愛機パピコン(某PC-6001のこと)にデータを落とすために作ったのですが、意外な御利益にありつけました。

また、ZVT.Xは一度に全データを読み込んでから処理を開始するので、十分なメモリがないと変換できません。現段階では変換するプログラムを自作するしかないようです。1992年6月号を見れば簡単に作れるでしょう(江藤啓さんに感謝)。

本プログラムはフリーソフトウェアとします。利用、配布など各自の責任の範囲で自由。

### <参考文献>

- Oh!X1991年12月号「冬の夜長のスペクトル解析」石上達也
- Oh!X1992年6月号「PCM8」江藤啓
- X68000ベストプログラミング入門、技術評論社



```

112: int ft()
113: {
114:     int i,sin1,cos1,amp1,sin2,cos2,amp2;
115:     short data;
116:
117:     sin1=cos1=sin2=cos2=0;
118:
119:     for (;counter<freq;counter+=rate) {
120:         data=getc(fp1);
121:         i=stblsize*counter;
122:         sin1+=data*sintable[(i /freq ) & stblmask];
123:         cos1+=data*sintable[(i /freq+pi2) & stblmask];
124:         sin2+=data*sintable[(i*2/freq ) & stblmask];
125:         cos2+=data*sintable[(i*2/freq+pi2) & stblmask];
126:     }
127:     counter-=freq;
128:
129:     amp1=pow2s(sin1)+pow2s(cos1);
130:     amp2=pow2s(sin2)+pow2s(cos2);
131:     return(amp1&amp;2?0:1);
132: }
133:
134: void casload()
135: {
136:     int i;
137:     short data;
138:
139:     i=1;
140:     data=0;
141:     while(!feof(fp1) == 0) {
142:         while(i-->0) data=data/2+ft()*0x0400;
143:         if ((i==data & 0x0601) == 0x0600) {
144:             data=(data/2) & 0x00ff;
145:             fputc(data,fp2);
146:             if (!info) putchar(data<0x0020?' ':data);
147:             i=1;
148:         }
149:     }

```

```

149:     } else {
150:         if (data == 0x07ff) {
151:             while(!feof(fp1) == 0) if ((data=ft()) != 1) br
152:         }
153:         i=10;
154:     } else {
155:         if (!info) putchar('\n');
156:         if (data != 0x0555) puts("サンプリングデータが
157:             break;
158:         }
159:     }
160: }
161:
162: void main(int argc,char *argv[])
163: {
164:     puts("Cassette Data Loader CASLOAD.X version 1.00 Copy
165:     cmdget(argc,argv);
166:     if ((fp1=fopen(fn1,"rb")) == NULL
167:         || (fp2=fopen(fn2,"wb")) == NULL) {
168:         puts("ファイルのオープンに失敗しました");
169:         exit(-1);
170:     }
171:     stblmake();
172:     counter=0;
173:     movetop();
174:     casload();
175:     fclose(fp2);
176:     fclose(fp1);
177: }

```

## リスト2 CASSAVE.C

```

1: /*
2:  * CASSAVE.C version 1.00 Copyright 1993 M.Harashino
3:  */
4:
5: #include <stdio.h>
6: #include <stdlib.h>
7: #include <math.h>
8: #include <ctype.h>
9:
10: #define stblsize 256
11: #define pi2      stblsize/4
12:
13: unsigned short freqtable[]={3900,5200,7800,10400,15600};
14:
15: unsigned int freq,rate,amp,header,counter;
16: int sign;
17: short sintable[stblsize*2];
18: char *fn1,*fn2;
19: FILE *fp1,*fp2;
20:
21: void help()
22: {
23:     puts( "使い方:CASSAVE <file name1> <file name2> [<avi
24:     puts( " -f<num>: サンプリング周波数設定 (default:1560
25:     puts( " -b<num>: ボーレート設定 (default: 120
26:     puts( " -a<num>: 振幅設定 (default: 100
27:     puts( " -h<num>: ヘッダ長設定 (default:
28:     puts( " -p : 位相反転
29:     puts( " -? : ヘルプメッセージ");
30: }
31:
32: void cmdget(int argc,char *argv[])
33: {
34:     int i;
35:     char c;
36:
37:     freq=4;
38:     rate=1200;
39:     amp=1000;
40:     sign=1;
41:     header=0;
42:
43:     fn1=fn2=0;
44:
45:     for (i=1;i<argc;i++) {
46:         c=argv[i][0];
47:         if (c == '-' || c == '/') {
48:             switch(tolower(argv[i][1])) {
49:                 case 'f':
50:                     freq=atoi(argv[i][2]);
51:                     break;
52:                 case 'b':
53:                     rate=atoi(argv[i][2]);
54:                     break;
55:                 case 'a':
56:                     amp=atoi(argv[i][2]);
57:                     break;
58:                 case 'p':
59:                     sign=-1;
60:                     break;
61:                 case 'h':
62:                     header=atoi(argv[i][2]);
63:                     break;
64:                 case '?':
65:                     help();
66:                     exit(0);
67:                     break;
68:                 default:
69:                     puts("無効なスイッチを指定しました");
70:                     help();
71:                     exit(-1);
72:                     break;
73:             }
74:         } else {
75:             if (fn1 == 0) fn1=argv[i];
76:             else fn2=argv[i];

```

```

77:         }
78:     }
79:
80:     if ( freq > 4 || rate > freqtable[freq] || amp > 32767 ) {
81:         puts("パラメータの指定に間違いがあります");
82:         exit(-1);
83:     }
84:
85:     freq=freqtable[freq];
86:
87:     if (fn2 == 0) {
88:         help();
89:         exit(-1);
90:     }
91: }
92:
93: void stblmake()
94: {
95:     int i;
96:
97:     for (i=0;i<stblsize;i++) {
98:         sintable[i]=sin(PI*2*i/stblsize)*amp*sign;
99:         sintable[i+stblsize]=sintable[i];
100:     }
101: }
102:
103: void bitset(int b)
104: {
105:     int i;
106:
107:     i=stblsize*(b>0?1:2);
108:     for (;counter<freq;counter+=rate) putw(sintable[i+count
109:     counter-=freq;
110: }
111:
112: void cassave()
113: {
114:     int i,c;
115:
116:     /* ヘッダを書き出す */
117:     for (i=0;i<header;i++) bitset(1);
118:
119:     while((c=fgetc(fp1)) != EOF) {
120:         bitset(0); /* Start bit (1bit) */
121:         bitset(c & 0x01);
122:         bitset(c & 0x02);
123:         bitset(c & 0x04);
124:         bitset(c & 0x08);
125:         bitset(c & 0x10);
126:         bitset(c & 0x20);
127:         bitset(c & 0x40);
128:         bitset(c & 0x80);
129:         bitset(1); /* Stop bit (2bit) */
130:         bitset(1);
131:     }
132:
133:     /* 終了コード 10101010101 */
134:     for (i=1;i<=11;i++) bitset(i & 0x01);
135: }
136:
137: void main(int argc,char *argv[])
138: {
139:     puts("Cassette Data Saver CASSAVE.X version 1.00 Copyr
140:     cmdget(argc,argv);
141:     if ((fp1=fopen(fn1,"rb")) == NULL
142:         || (fp2=fopen(fn2,"wb")) == NULL) {
143:         puts("ファイルのオープンに失敗しました");
144:         exit(-1);
145:     }
146:     stblmake();
147:     counter=0;
148:     cassave();
149:     fclose(fp2);
150:     fclose(fp1);
151: }

```

## 猫とコンピュータ

## ゴメンナサイの値段

Takazawa Kyoko

高沢 恭子

思いがけないトラブルやアクシデントは避けられないものですが、そのことによる被害よりも、ほんとうに重要なのはそのあとかもしれません。あやまることも許すことも難しくはないはずですが……。

恐怖の注意書きを説明書の後半に見つけた。「この接着剤は皮ふを瞬間に、強力に接着するため、使用に際して十分注意すること」。もう、くっついてしまった左手の親指と人さし指を、悲鳴とともにひきはがしたあとだった。「あやまって指などを接着したときは、無理にはがさず、お湯の中でゆっくりもみほぐしながらはがすこと。または◇◇ボンドはがし液を使用すること」だそう。

指がくっついたまま、はがし液を買いにいく姿が浮かんできた。

コワくて便利なものを3つあげなさいといわれたら、これからは瞬間接着剤をいちばんにしよう。危険率からいえばクルマやヒコウキの比ではない。使っているうちに筆算がだんだんできなくなる計算機も、知らないうちに漢字が書けなくなっていくワープロも、もうコワくない。

## あぶない集まり

瞬間接着剤を使ったのは、自転車のライトを修理するためだった。

スーパーマーケットの自転車置場で、私の自転車はなにかの理由で倒れたらしい。そのときライトのカバーが根元からポッキリ欠けてしまって、細い配線コードだけであやうくぶらさがっていた。

自転車はたくさん集まると、ハンドルやペダルがからみあって、まるでカマキリの集団のようだ。おたがいの動きをあぶなくする。どれも刺激しないで、そおと1台

だけをとりますのは、むずかしい。

それでもすこし雑然と置かれているほうが、まだあつかいやすい。駐輪場が込みあつてきて、係員が自転車をすきまなく並べなおすと、お客は不便になる。人の入る余地をつくってくれないので、買い物の荷物でボリュームの増した人たちは、カマキリの隊列にわりこむのにとて苦勞する。自転車の将棋倒しも、整列させてあるときのほうが広範囲になるように思う。

自転車なんて、そんなふうに毎日倒れたりキズついたりしているものだ。誰が見ても古びた白い自転車が、ちょっとしたハズミで倒れてライトが欠けてしまっても、びっくりする人はいないだろう。

でもやはり、倒した人は知っていたのかなとつい考える。それほど人も多くなかった午後の早い時間だった。私なら持ち主があらわれるのを待って、せめてあやまることくらいするのにな、なんて思う。

ばかばかしい。自転車くらいのことでもわざわざ時間をついやして詫びる人なんているものか。倒した人だって、ライトに気づかなかったかもしれないし、小さな子供がしたことかもしれない。と思いながらもなにかさみしさがあるのは、やっぱりゴメンナサイがほしいのだろうか。

## ことわざ遊び

子供のころ父や兄の本棚から本をぬきだして読むのは、けっこう楽しい遊びのひとつだった。そのなかでも、ことわざ辞典の

一群にはふつうの本とはちがう、わくわくするようなものがあった。「金言名句全集」「故事成語辞典」「西洋俚諺集」といったタイトルのそれらの本には、おとなたちがほんとうに考えているのはこんなことだという秘密が、かたはしから並べてあるような気がした。

それも、世界じゅうの人たちが長い年月かけてつきとめたるし教訓や悪知恵、本音のエキスが、つきからつきへと書いてあった。物語のように長い道のりをたどって1つの感慨をさずかるのではなく、2行、3行でテーマだけがズバリとわかるのだから、効率のよいよみのだった。

そのころ、「格言や名言のようなものは、その調子のよさにのせられて、ものごとをいかげんところで解決させたり、考えを上滑りさせたりするものだ」という意見を讀んだ。まったくそのとおりだと思いながら、やっぱりその調子のよい節まわしを知っていることがうれしかった。

とくに小説や文学、聖書からの抜粋、哲学者たちのことばは、うたがわしいところがかえって魅力だった。

「○○は大きなあやまちは許すが、小さなあやまちはこれを許さない」。

自転車のライトをこわされて苦々しく思っているとき浮かんできたのが、この一文だった。「○○」に入っていたのは「男」「女」「人」のうちの1つである。

格言や名言といわれるものにふれたとき、多くの人は「ふーんなるほど」と同意しながら、「ほんとうにそうだろうか」と、それを裏づけるため、あれこれ思い浮かべてみるものだ。

上の「○○」に3つのうちのどのことばが入るか不明だとしたら、どれが入ることでの的中し、それが名言といえるものになるかについて、迷いがあるだろう。そして、順に主語を入れかえてみると、意味あい微妙に変わってゆくのがわかる。「大きなあやまち」とは、たとえば戦争や殺人、職務の上の不正なのか。「小さなあやまち」とは、たとえばだいじんグラスをこわされたり、自尊心を傷つけられたりすることなのか。

主語を入れかえてみて、ついでに大小も入れかえてみると、「名言」の主張はほんどうたがわしくなる。



## 1円玉が残るとき

3つの単語に加えて、「コンピュータ」を主語にしてみたらどうなるか。

当然、大きなあやまちも、小さなあやまちもゆるしていないと、誰でも思う。

人間が「ゆるさない」ときは、相手に謝罪させたり、賠償をもとめたりということもあるが、コンピュータが「ゆるさない」ときは、怒りの要素はない。受け入れるか否か、応答するかしないかで、その結果、正しいことだけが、しごととして残ると思われているらしい。

先日ある人から、とてもめずらしいフシギなできごと、というのを聞いた。ソフトで計算した社員の給料を、じっさいに分配する段階で、どうしても1円だけ余りが出てしまうという、給料計算の怪だ。

その会社では社員の希望から、従業員の給料は銀行振込ではなく、本人に手渡す方式がとられているそう。毎月、1人ひとりの給料袋に、どの種類の紙幣が何枚いるか、硬貨がそれぞれいくつずつあればいいかを、ソフトで割り出しておく。その計算にもとづいて、銀行に準備してもらったお金を正しくわけていけば、完了したときはすべての貨幣が袋におさまっているはずである。

ところが、なぜか1円だけ残る。いくらチェックをくりかえしても計算のあやまちが見つからない。原因はけっきょくわからずじまいで、むろん、誰からも苦情はなかったという。

コンピュータはきまりどおりに働くけれど、ときどきフシギなしごとを残すこともある、というのも答えの1つらしい。

すこし事情は異なるけれど、外国の文学などを翻訳ソフトを使って、日本語に書きかえてみたとする。1つひとつ正確に翻訳をほどこしていき、どこにもあやまりがないものができあがる。しかし、作品全体の姿はどこか原作からかけ離れている、といったことは起こらないだろうか。

そういうときのコンピュータは、大きなあやまちはゆるすけれど、小さなあやまちはゆるさないということになりそうだ。

## 巨人軍来たらず

「〇〇」のなかは「女」だった。

これを言った人（たぶん男性）にとって、女はものごとの軽重を考えず、他人のささいな過失はゆるさないくせに、戦争や政治などのあやまちはあっさりあきらめてしまうもの、だったのだろう。

そういえば、こんな被害を体験した。

13時間にわたって東海道、山陽新幹線をマヒさせ、30万人に影響を与えたJR発足以来最大規模のトラブル。その日の早朝に東海道新幹線の浜松で、保線作業車同士が追突、脱線したためだった。

8月はじめ、夏休みの週末とお盆前の帰省ラッシュが重なった時期のこの事故は、余波による二次的な迷惑の大きさでも記録的だった。私もこの日10時間あまりを、事故の流れといっしょにすごした。

東京正午発、名古屋までの指定券を用意していた。名古屋からは近鉄特急で、夫のいる三重県上野市までいく計画だった。

早朝のテレビで事故を知り、11時すぎには東京駅に到着、待機。じっさいに運転が再開されたのは午後5時すぎだったが、3時までを中央地下道ですごした。

この日の事故の影響が際限なくひろがった原因の1つは、復旧がますます悲観的になっていくにもかかわらず、情報がゼロに近いまいたずらに時間がたち、乗客数がふくれる一方だったことだ。

午後2時くらいからは、「本日の旅行はお取りやめをお願いします」とアナウンスが流れはじめた。泣き顔の子供たちをなだめながら家に帰る人もいたが、休暇をとった人、ビジネスが目的の人などはやるわけにいかない。「こだま」だけが1時間に1本くらいは動いていたので、無謀にも3時20分ころの「こだま」で静岡へ。地獄のような混雑のなか4時半着。あとは東海道線しかない。ところが運よく「のぞみ」が名古屋まで走るようになった。発車まで1時間待って5時半発、6時半名古屋着。

近鉄特急は全席指定だが、もちろん権利は喪失していた。検札の車掌さんに事情を説明し、あいていた席にすわらせてもらう。この日はじめて得た座席だ。「あのお、新幹線の事故ってほんとだったんですね」となりの席にいた若い女性が話しかけてきた。「巨人—中日戦を見に名古屋球場までいったんですけど、巨人軍の選手がこれなくなって中止になったんです」

巨人軍カラーのヴァーミリオン（朱色）のポロシャツを着た彼女は、バッグをあけてメガホンと手製の紙吹雪を見せてくれた。「みんなすごく怒ってました」と彼女。

午後8時半には夫のマンションに到着できたが、この1日の新幹線のマヒが日本じゅうの人たちに与えた被害は、歴史的であったそう。海外出張やバック旅行の予定をすべてダメにした人。メインゲストを失ってしまった、さまざまなイベントや講演など。主催チームが売り上げのすべてをとるプロ野球で、ドル箱試合をフイにした中日球団の損害は莫大だそう。

どんなことをゆるせないと思うかで男女にハッキリちがいがいるかどうか、よくわからない。ただ、被害を受けた人が多いほど、いたみは分散されることはたしかだ。これを「あきらめ」という人もいる。だから私は、政治の腐敗よりも、だいたいな画集をよごされたほうがぐやしい。

帰路の新幹線名古屋駅で、特急券の払い戻しを受けた。座席指定ぶんの料金だけかと思っていたら、特急料金の全額4,610円が戻された。でも、なんのこともつげ加えてもらえなかった。

もともとこちらが支払ったぶんを返してもらいよりも、ひとことの「ゴメンナサイ」がほしかったのに。

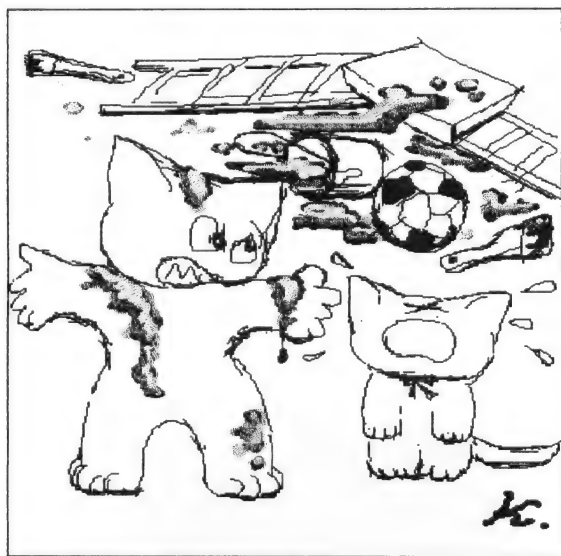


illustration : Kyoko Takazawa

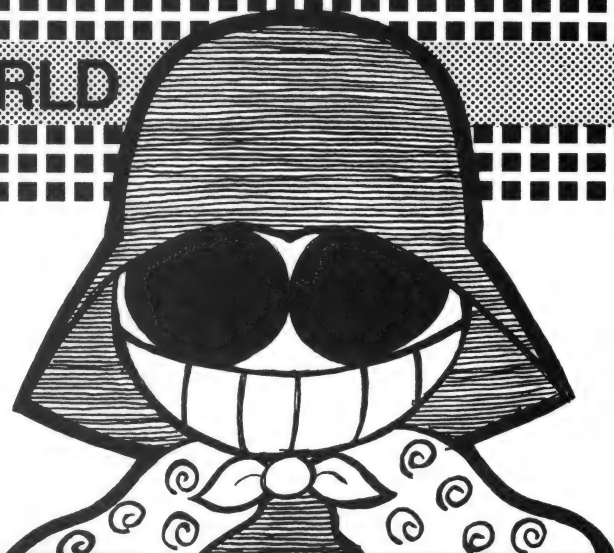
# ANOTHER CG WORLD

## 今回のCGデータ

総物体数 91 メタボール数 8

使用ソフト キャラクタ：C-TRACE 岩：サイクロン  
岩の軌跡：MATIER

背景の岩はテクスチャマッピングとバンプマッピング  
を施している



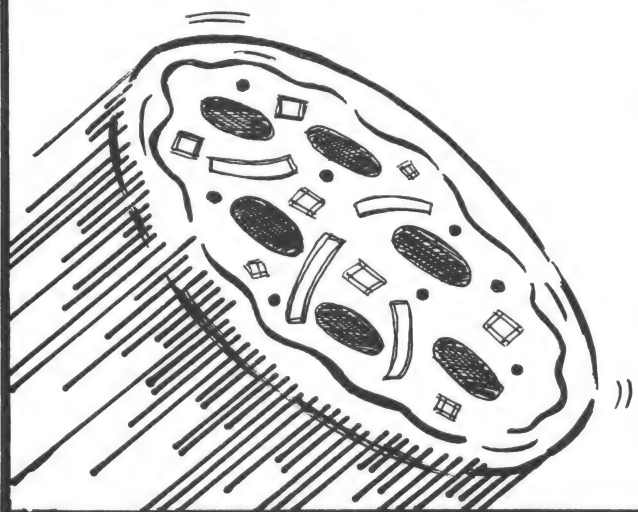
© 1993 K&K

スター  
ウォーズの  
シニア・ファルコン号は

ジョージ・ルーカスが...



ピザを食べているときに

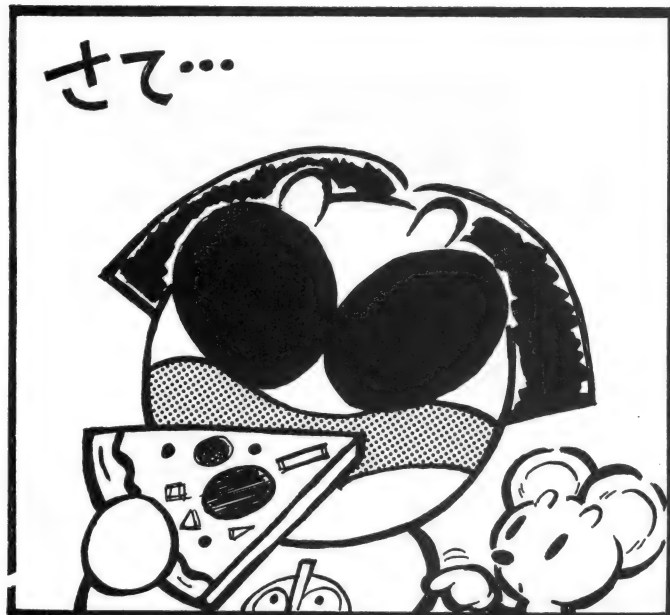


デザインを思いついたんだ  
そう...

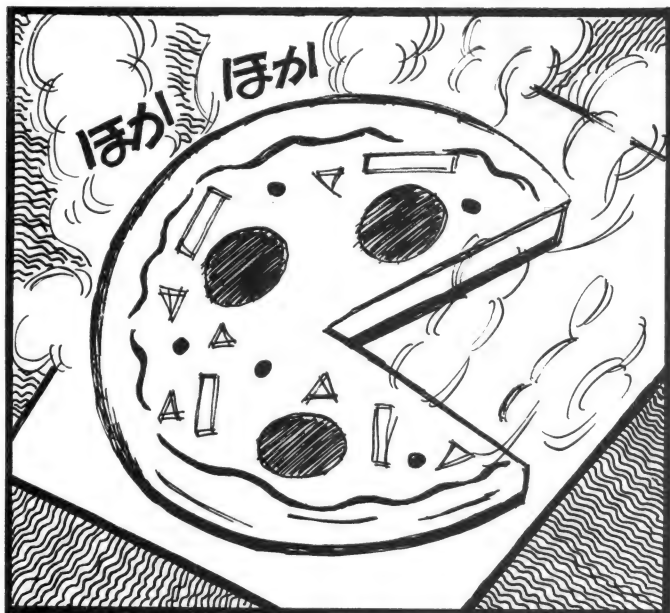




さて...



これか!  
何かに見えないか?って



ふうむ



また、食べもののおはなしですみません



「ジョージ・ルーカス展」  
スターウォーズやエイリアンズのSFXで  
使われた 模型や マットペインティングが  
あって とっても楽しい。東京は、終わ  
っちゃったけど...

名古屋 松坂屋 12/4 ~ 29  
博多 大丸 '94 1/4 ~ 18  
熊本 鶴屋百貨店 3/19 ~ 4月上旬  
ひろしま 現代美術館 4/16 ~ 5/29  
大阪 梅田 大丸ミュージアム 8/3 ~ 8/27  
映像制作にキョーミのある人は一見の価値アリ

## NEW PRODUCTS

### 5.6型液晶カラーテレビ 6E-C3/DK3 シャープ



6E-C3

シャープは5.6型液晶カラーテレビ「6E-C3」、ダイバーシティカーキット「6E-DK3」を発売した。

「6E-C3」は新開発の5.6型低反射・高輝度TFT液晶を採用することで、従来の「6E-C1」シリーズに比べて外光反射が約1/10、輝度が約2倍になり、見やすくなった。さらに電波の弱い地域でも自動的に受信状態を高めるオートブースター機能により受信感度が3～6dB向上し、受信エリアが約20%拡大する（同社の調べによる）。

「6E-DK3」のキット内容は、

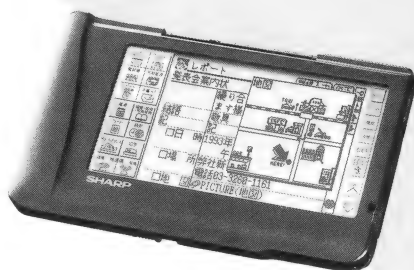
- ・液晶カラーテレビ本体「6E-C3」
  - ・4入力ダイバーシティステーション
  - ・クイックリリーステレビスタンド
  - ・ワイヤードリモコン
  - ・ダイバーシティカーアンテナ 2本
- で、スタンドはネジ留めをする必要がなくワンタッチで着脱できる。

価格は「6E-C3」が100,000円、「6E-DK3」が144,000円（ともに税別）となっている。

〈問い合わせ先〉

シャープ(株) ☎043(299)8210,06(621)1221

### 新携帯情報ツール 液晶ペンコム PI-3000 シャープ



PI-3000

シャープはポケットサイズの新携帯情報ツール液晶ペンコム「PI-3000」、愛称ZAURUS（ザウルス）を発売した。

特徴としては、手書き文字認識によりメモ感覚で入力したデータをそのまま利用できるレポート作成機能、作成したレポートをタイトルごとに見やすくするファイリング機能がある。さらにPIM機能では仕事に優先順位をつけられるアクションリスト、ひとつのキーワードに関するデータをピックアップするアクションプランナーをはじめ、カレンダー、スケジュール、名刺管理などが行える。ほかにも国語/英和/和英の3冊の辞書機能、「PI-3000」や日本語ワープロ「書院」、パソコンなどとのデータ交換が可能な光通信機能がある。

また、ハイパー電子システム手帳「DB-Z」用のICカードが使える。

価格は65,000円（税別）。

〈問い合わせ先〉

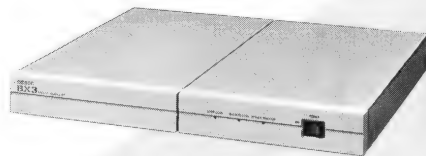
シャープ(株) ☎043(299)8210,06(621)1221

### 超薄型無停電電源装置 BX3 オムロン

オムロンは超薄型無停電電源装置「BX3」を発売した。

本機は回路部とバッテリー部を別にした

BX3



2ボックスタイプの無停電電源装置である。出力容量は300VAで、パソコン装置1台程度なら5分間のバックアップを行う。

従来、バッテリーの交換はメーカーが行っていた。それをバッテリー部を別にすることで、ユーザーが自分で交換用バッテリーユニット（別売）を購入して取り替えるだけで、メンテナンスが行えるようになった。また、従来機種同様、停電・過負荷・バッテリーローの3種類のアラームを標準装備し、ブザーとLED表示で知らせる。

大きさは190mm（幅）×335mm（奥行）×41mm（高さ）と小さく、重さも2ボックス合わせて5kgと軽い。設置方法も縦置き、横置き、重ね置きなど使用環境に応じて選べる。

価格は42,800円（税別）。

〈問い合わせ先〉

オムロン(株) ☎03(5488)3221,06(282)2672

### 電子辞書 TR-245/345 セイコー電子工業



TR-245



セイコー電子工業は電子辞書「ICディクシヨナリー・ポケット」シリーズ、漢字辞書「TR-245」、英和・和英辞書「TR-345」を発売した。

「TR-245」はローマ字で読み方を入力し、変換キーを押すだけで画面に漢字を表示する。収録単語数は約38,000語。同音異義語がある場合には画面にマークを表示。

「TR-345」は英単語・日本語の単語を入力し訳キーを押すだけで画面に日本語訳・英訳を表示する。どちらの場合も、見出し語約11,000語に対し、訳語約19,000語を収録している。

両機種とも電卓機能を搭載し、2枚貝のように開閉するシェルタイプの外観で、携帯時の保護が配慮されている。大きさは閉じた状態で108mm(幅)×67mm(奥行)×12mm(厚さ)と片手に収まるサイズである。

価格は「TR-245」「TR-345」とともに4,500円(税別)。

<問い合わせ先>

セイコー電子工業(株) ☎0120(052)440

### パーソナルワープロ HW-9900RX カシオ計算機



HW-9900RX

カシオ計算機はパーソナルワープロ「HW-9900RX」を発売した。

本機は960×600ドットの高精細液晶を表示画面に採用し、標準表示で1文字あたり24ドットの表示となる。表示文字は16,12ドットに切り替え可能で、B4判の横置きも12ドットで一覧表示ができる。

図形そのものを自在に変形できるハイパーグラフィック機能、イラストやデザイン画の作成を助ける約300種類の図形ライブラリーなどでグラフィック機能を強化している。ほかにも従来からある、文字列を自在に変形するハイパーアウトライン、用紙

の特性に合わせて自動レイアウトする自動編集などの機能を搭載。

書体も和文7書体、欧文19書体をスーパーアウトラインフォントで標準装備している。そしてマルチテキストコンバータ機能で、他社の機械で作成した文書も読み込めるようにした。

価格は238,000円(税別)。

<問い合わせ先>

カシオ計算機(株) ☎03(3347)4811

### フォトビジョン FV7 富士写真フイルム



FV7

富士写真フイルムはハンディタイプのフジックスフォトビジョン「FV7」を発売した。

本機は、35ミリのネガフイルムやスライド/写真/印刷物/立体物などを、簡単にすぐテレビで見ることができる。レンズには高解像度41万画素CCDのフジノンレンズを搭載し、シャープで高画質な映像が得られる。出力端子はS映像出力、ビデオ出力の各端子をひとつずつ装備している。

また、フジックスデジタルイメージファイル「DF-10」と組み合わせれば、画像入力機として、パソコンなどを介さず、デジタル静止画像を3.5インチフロッピーディスクに記録できる。

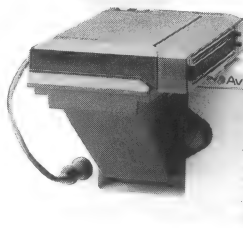
大きさは収納時は104mm(幅)×62mm(高さ)×214mm(奥行)と非常にコンパクトで、重さは約700g。

価格は74,800円(税別)。

<問い合わせ先>

富士写真フイルム(株) ☎03(3406)2111

### パーソナルフイルムレコーダ “写嬢” FR-1300 日本アビオニクス



FR-1300

日本アビオニクスはパソコンの画面イメージを各種フイルムヘブリントするフイルムレコーダ“写嬢シリーズ”「FR-1300」を発売した。

特徴としては水平走査周波数が21~70kHzのディスプレイにマルチスキャン方式で対応し、1670万色のフルカラーに対応している。接続もアナログRGB信号のため、ドライバソフトは一切必要ない。

ほかには、オートコントラスト/オートブライトネス機能を搭載している。

価格は298,000円(税別)。

<問い合わせ先>

日本アビオニクス(株) ☎03(3725)3814

## INFORMATION

### ヒューマンクリエイティブスクール・ エンタテインメント・スピリッツ'93 ヒューマンクリエイティブスクール

ヒューマンクリエイティブスクールは“ヒューマンクリエイティブスクール・エンタテインメント・スピリッツ'93”と銘打った学園祭を行う。

これはアミューズメント産業を目指す人の学校の学園祭として、多くの人が楽しめるエンタテインメント性の強いものを目指している。これに併せて第1回ヒューマン・コンピュータ・エンタテインメント・コンテストのオープニングイベントが実施される。

開催日は1993年11月13日(土)、14日(日)の2日間。場所は武蔵野東小学校。

<問い合わせ先>

ヒューマンクリエイティブスクール

☎0422(23)1111

シャープとアップルコンピュータの大型提携商品である超小型端末機が、いよいよデビューだそうだ。

この機械に関しては、新聞にごくごく簡単に書いてあった以上のことはまったく知らないが、おそらく電子手帳よりもパソコンっぽい小型携帯コンピュータなのだろう。そういつてしまえば、あちこちのメーカーからあれこれと発売されているものをイメージしてしまう。だが、それなりに使えるソフトをICカードで手軽に扱えるようにしたただけのことで電子手帳を爆発的なヒット商品に仕立てあげたシャープが、独創力抜群のアップルと組んで作ったのだから、そこの商品とはちょっと違うことだけは間違いない。見た目はそう変わらなくても恐るべき商品となる可能性は十分あろう。

さて、これに限らず、このところSFの小説や映画でよく出てきそうなアイテムのひな型が順々に生み出されつつある。

MD(ミニディスク)なんかそうだし、松下電器が先日発表したフラットディスプレイテレビなんかも、SFではおなじみ。

小型通信機の仕掛けも、今年はいろいろと始まっている。世界中を66個の人工衛星で結んで国際・国内電話ともかけられる小型携帯電話普及プロジェクト「イリジウム計画」が始動したし、10月にはこれとはまったく別に、日本国内で考えられている新しい携帯電話機「PHP(パーソナルハンディホン)」の実験も始まる。

小型通信機といえば、SFや特撮の必須アイテムといえるのが、腕時計型のテレビ電話機。数年前にモトローラが腕時計ポケットベルを発売していることもあり、そう遠い話でもないのだろう。

ちなみに、こうした「SF製品」のひな型が次々と発表されている理由をちょっと考えてみよう。

まずありきたりの理由だが、テクノロジーがその水準に達しつつあること。ICメモリも16MビットのダイナミックRAMが量産体制に入り始めているし、i386クラスのプロセッサとか32ビットのDSP(デジタル信号処理プロセッサ)とかがそこのゲートアレーに組み込まれるようになってきている。こうなると、小さな機械にも相当の機能が詰め込める。

だが、こうした性能面だけの話ではないとばかりは思う。そろそろハイテク商品も「出

尽くし状態」に近づいて、話題がOSだのプロトコルだのという、見えない部分にのみ集中する時代が続いていた。こうなると、消費者としては飽きがくる。

単純なマイナーチェンジ商品では、飽きられて買われなくなるのは当然のことだ。ましてや不況による消費低迷時代。よほど凝った商品を生み出さないと、ヒットは期待できないということをメーカーもようやくわかってきたのではあるまいか。

ところで、「SFによく出てくる」みたいな表現を使うことはたやすい。だが、マニアやファンの人を別にすると、我々ほどの程度、SFになじんでいるのであろうか？

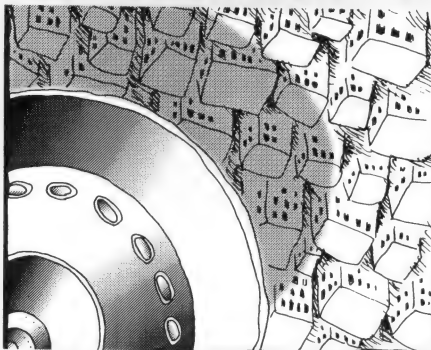
まず映画でチェックしてみたい。

## X - OVER・NIGHT

(クロスオーバーナイト)

### [第40話]

## SF時代に向けて



TAKAHARA HIDEKI 高原 秀己

本棚に入っていた「外国映画ベスト200」(角川文庫)のSF映画人気ランキングを紹介してみよう。これは、芸術性とか価値とかは度外視した単純な人気投票をまとめた紹介本である。

- (1)2001年宇宙の旅
- (2)スター・ウォーズ
- (3)E.T.
- (4)キングコング (1933年版)
- (5)未知との遭遇
- (6)エイリアン
- (7)ミクロの決死圏
- (8)ブレードランナー
- (9)猿の惑星
- (10)ジョーズ

サスペンスである「ジョーズ」がSFのところにランキングされているのには首をひねるが、それはともかく、おそらく旧キングコング以外はすべて見ている人が多いと思う。そう、SF映画は繰り返しテレビでオンエアされているのだ。しかもPart 2, 3と次々と量産されることもあり、まったく見ないほうが難しい状況ですらある。

この本のSF映画10位以下で毛色が変わったところでも、(14)惑星ソラリス、(20)時計じかけのオレンジ、(26)博士の異常な愛情、(27)未来世紀ブラジル、(40)未来惑星ザルドス……といった程度であるから、われわれはかなりSF映画に関しては造詣が深いといつてよいのかもしれない。

ところがSF小説については、ばくもそうなのだが、意外なほど目を通していないのではなかろうか。純文学やミステリーと違って、SFの場合「これだけは読みましょう」的な情報が紹介されているようで、されていないような気もする。あと、ハヤカワ文庫にしても創元推理文庫にしても、外国の作品はかなりぶ厚くて細かい字が山のように並んでいるので、圧倒されてしまってなかなか読み進めない辛さもある。

とはいえ、アシモフ、クラーク、ハインラインらの巨匠による著作、コナンシリーズや火星シリーズなど超有名なものは、機会があれば、いくつかは読んでおきたい。

「ハイテクを語るうえでのSF」という視点では、こうした古典的名著以上に押さえておきたいのが、小松左京と星新一の著作。特に、読んでいない人が意外と多い小松左京の中・短編集(新潮文庫など)のなかには、時代を予見したようなSF描写が極めて多い。どちらかという超常現象は小松左京が、メカものは星新一が強いといつていいかもしれない。たとえば、松下のフラットディスプレイテレビの話をしたが、「トータル・リコール」の世界にとどまらず、星新一の短編には「テレビシート」なるものまで登場しているのだ。

あとはファンタジー系小説の地位がぐんぐん高まっているので、こちらも少しは読んでおきたいところ。スペースもなくなったので、神月摩由璃の著書「SF&ファンタジーガイド」(教養文庫)をお薦めして終わりにする。「指輪物語」「ゲド戦記」から国産ものやコミックまでかなり幅広く紹介してあるのが特徴だ。



郵便はがき

1 0 3 - 0 0

1 6 1

料金受取人払

日本橋局承認

1564

差出有効期間

平成 7 年 5 月

14日まで

(受取人)

東京都中央区

日本橋浜町 3-42-3

ソフトバンク株式会社

**Dhl**  編集部行

キ  
リ  
ト  
リ  
線

□□□ - □□

電話

住所

フリガナ

氏名

年齢

職業・勤務先  
学校・学部・学年

今月号の特集について

いちばん良かった記事

興味のなかった記事

これから載せてほしい記事内容

本誌以外にお読みのパソコン雑誌

期待している新作ソフト：

推薦理由：

最近買って気に入ったソフト：

推薦理由：

あなたがいちばん好きなゲームのジャンルはなんですか？

あなたの愛機は(所有機種に○印をつけてください) ない

X1(マニアタイプ,C,D,F,G,twin) X1 turbo(model 10,20,30,40, II, III, Z, ZII, ZIII)

MZ-(80K/C, 1200, 700, 1500, 80B, 2000, 2200, 2500, 2861)

X68000(初代,ACE,PRO,PROII,EXPERT,EXPERT II,SUPER,XVI,Compact, HD)

X68030(CZ-500/510,300/310) その他 MIDI楽器( )

FD( 基) TAPE QD HD( MB) MO プリンタ( )

年齢 歳 パソコン歴 年 男・女 プレゼントNo.

キリトリ線



# 振替用紙

◆点線からきれいに切り取ってご使用ができます。

通常払込料金  
加入者負担

払込票

東京	1	29307
ソフトバンク株式会社		
加入者名	全額	

払込人住所氏名	備考
受付局日付印	

記載事項を訂正した場合は、その箇所に訂正印を押してください。

通常払込料金  
加入者負担

通知票

東京	1	29307
ソフトバンク株式会社		
加入者名	料主	備考
払込人住所氏名	郵便番号	

受付局日付印
--------

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。（郵政省）

各票の※印欄は、払込入において記載してください。

www.elsevier.com/locate/jmb

© 2006 Blackwell Publishing Ltd, *Journal of Internal Medicine* 260: 103–111

100

＜例＞ 4月11日発売 (Oh! PC 4月15日号) の

問に合わなかった場合は、自動的に次の送金に含めさせていただきます

● 2006年11月11日は「世界発光日」より連  
りますので「水卜さい」

THE WINSTON-SALEM  
SOCIETY OF MUSICIANS  
AND THE  
SOCIETY OF MUSICIANS

MAGAZINE

Oh ! Dvni

# Index

（一）2011年11月21日

2019年12月10日

の<sup>※</sup>は、加入者あての通信にお使いいただき

定期購読			
Oh! PC	年部 (2200)	12,560円 (税込)	部誌 NO.
Oh! PC	6 = 11 (1200)	6,720円 (7折)	部誌 NO.
Oh! X	年部 (1200)	7,200円 (9折)	部誌 NO.
Oh! FM TOWNS	年部 (1200)	7,440円 (9折)	部誌 NO.
C MAGAZINE	年部 (1200)	7,680円 (9折)	部誌 NO.
Oh! Drive	年部 (1200)	8,100円 (9折)	部誌 NO.
月刊 PC	0 10 (1200)	7,680円 (9折)	部誌 NO.
月刊情報処理技術	年部 (1200)	9,360円 (9折)	部誌 NO.
月刊情報処理技術	6 = 11 (600)	4,080円 (9折)	部誌 NO.
LANTIMES	年部 (1200)	7,760円 (9折)	部誌 NO.
THE WINDOWS	年部 (1200)	7,680円 (9折)	部誌 NO.
Windows Magazine	年部 (1200)	9,360円 (9折)	部誌 NO.

この私込通知書は、機械で使いますので、下部の欄を汚さないよう特に御注意ください。また、本書を折り曲げたりしないでください。(郵政省)



## プレゼントの応募方法

とじ込みのアンケートはがきの該当項目をすべてご記入のうえ、希望するプレゼント番号をはがき右下のスペースにひとつ記入してお申し込みください。締め切りは1993年11月18日の到着分までとします。当選者の発表は1994年1月号で行います。また、雑誌公正競争規約の定めにより、当選された方はこの号のほかの懸賞には当選できない場合がありますので、ご了承ください。

2

EAビクター  
☎03(5410)3111

コットン

X68000用 5"2HD版  
9,800円(税別)



3名

主人公はお気楽だけど、ゲームの難易度は……。人気のおかげで、続編も企画されています。

4

ソフトバンク  
☎03(5642)8100

レーザー  
アクティブの  
すべて

1,600円(税込) 5名

話題のAVマシン「レーザーアクティブ」の魅力を徹底解剖したSOFT BANK MOOKです。



1

ネクタイピン



1名

岐阜県の坂井秀昭さんよりお送りいただいたネクタイピン。なんと、そこには「OHX」のアヤシイ3文字が……。坂井さん、ありがとうございました。

3

EAビクター  
☎03(5410)3111

湯飲み



非売品

5名

10月号で好評だったので追加プレゼントです。ちなみに2種類ではなくて、コットンちゃんの反対側に「寿」と書いてあるのです。

## 9月号プレゼント当選者

1 餓狼伝説 (静岡県) 福井知幸 (京都府) 国政 寛 (山口県) 岡井正和 2 ロボットコンストラクションR.C. (北海道) 堀井 晶司 (埼玉県) 長崎 望 袴田 健 3 68COLOR JOY CONT TurboV (新潟県) 吉田晴彦 (埼玉県) 鈴木正人 (神奈川県) 塚田矩旭 (大阪府) 島誠一郎 (鳥取県) 堀尾忠教 4 テレホンカード (千葉県) 大橋隆雄 (東京都) 小倉圭司 高橋 明 (大阪府) 幸 俊威 (香川県) 西原圭一 5 にこっ! (千葉県) 星野こずえ (埼玉県) 松木明博 (三重県) 中山剛志 (兵庫県) 村瀬正美 (福井県) 平木敬太郎 (敬称略) 以上の方が当選しました。商品は順次発送いたしますが、入荷状況などにより遅れる場合もあります。

# FILES

## Oh!

このインデックスは、タイトル、注記——著者名、誌名、月号、ページで構成されています。寒くなってきましたね。そろそろ冬の貌が見え隠れしてきました。風邪などひいて、寝込まないように健康には気をつけてね！

### 参考文献

I/O 工学社  
ASCII アスキー  
コンプティーク 角川書店  
C Magazine ソフトバンク  
テクノポリス 徳間書店  
電撃王 主婦の友社  
POPCOM 小学館  
マイコンBASIC Magazine 電波新聞社  
My Computer Magazine 電波新聞社  
LOGIN アスキー

## 一般

### ▶THE NEWS FILE

最新第三世代AVマックシリーズ発表、驚異のグラフィカルマシンINDY登場、トヨタオートサロン「あむらっくす大阪」の紹介など、パソコン関連の製品・イベント情報コーナー。——編集部, LOGIN, 18号, 26-33pp.

### ▶CG最前線 Part 1

映画やCM、イベントなどに進出しつつあるCG。その仕掛人のひとりであり、YMO再生コンサートのCGを担当した原田大三郎氏ほかにインタビューする。——編集部, LOGIN, 18号, 196-201pp.

▶MUSIC LABO コンピューターミュージックのススメ  
MIDIで音楽を楽しむための情報を紹介。ほかにはコンピュータミュージックのルーツを解説したり、LOGIC SYSTEMの松武秀樹のインタビューがある。——編集部, LOGIN, 18号, 202-207pp.

### ▶電網幼稚園

初心者ネットワークのための教育機関。今回は初心者が間違いやすい失敗をQ & A方式で紹介する。——編集部, LOGIN, 18号, 232-233pp.

### ▶THE NEWS FILES

第31回アミューズメントマシンショーの模様、エプソンのパソコン最新機種、電気睡眠導入機などのハイテク関連グッズとイベントの情報。——編集部, LOGIN, 19号, 28-35pp.

### ▶CG最前線 Part 2

放送業界などでの最新のCGの動きを紹介する。テレビやイベントで使われているCGをチェックし、CGアーティストの新城欣一氏にインタビューを行う。——編集部, LOGIN, 19号, 206-211pp.

### ▶電網幼稚園

FAXモデムについて教える。本当に役に立つのかどうか、使い方に関する疑問を解明。おすすめモデムの紹介つきた。——編集部, LOGIN, 19号, 246-249pp.

### ▶電撃アニメーション

日本独自の発展を見せたTVアニメーション。いわゆる「アニメ」が世界でどう受け取られているか、アメリカのアニメコンベンションのレポートなどを交えて紹介する。——編集部, 電撃王, 10月号, 156-165pp.

### ▶特捜情報最前線

シャープのカラー液晶ディスプレイや、プロサイドの激安14400bpsモデムなどの製品情報。ほかにはソフトのトップセールス20やアーケードゲームなどの紹介もある。——編集部, コンプティーク, 10月号, 19-31pp.

### ▶EXPO'S IN USA SUMMER '93

ゲームにもかかわりのある「MACWORLD EXPO」と「MULTIMEDIA'93」の模様をレポートする。——編集部, コンプティーク, 10月号, 32-35pp.

### ▶NEWS CLIP

NECパソコンアートフェスティバルの模様のレポート、TEPIA、フジタヴァンテといったハイテクアミューズメントスポットの紹介など、パソコン関連の話題。——編集部, POPCOM, 10月号, 21-26pp.

### ▶新鮮良品館

デジタル録音・再生の楽しみで大人気のDCCとMDの最新製品を一挙に紹介。その他ミニコンボや格闘ロボットおもちゃの新製品など。——編集部, POPCOM, 10月号, 118-119pp.

### ▶今月の新製品ピックアップ

パソコンからコントロールできる8mmビデオデッキ「CVD-500」や9600bps低価格FAXモデム「FAX MP96」など、パソコン関連の新製品を紹介するコーナー。——編集部, マイコンBASIC Magazine, 10月号, 42-43pp.

### ▶パーソナル・ページ・プリンタ時代到来！

業界最低価格を実現したCASIO「ページ即写mini」とコンパクトさが売りのNEC「PC-PRI000E/4」をレポート。両者のクオリティと買い得度をチェックする。——編集部, マイコンBASIC Magazine, 10月号, 62-65pp.

### ▶新製品Flash NEWS

満開製作所のX68000/030シリーズ用増設5インチFDDやシャープのカラー液晶ディスプレイなどの新製品を紹介する。——編集部, マイコンBASIC Magazine, 10月号,

78-82pp.

### ▶Bug太郎のプログラム・タイム その10

「熱血格闘宣言！(後編)」と題して、格闘ゲームのアルゴリズムを考える。多関節のキャラクターが動き回るゲームを実際に制作。——谷裕紀彦, マイコンBASIC Magazine, 10月号, 88-89pp.

### ▶先生と生徒のためのBASICプログラミング講座

ゲーム制作・教材制作に役立つテーマを取り上げて解説する講座。テーマは浮力。浮力を使ったバルーンゲーム作りに挑戦する。——東幸太, マイコンBASIC Magazine, 10月号, 96-100pp.

### ▶そろそろCD-ROMドライブがほしい

これからCD-ROMドライブを買うという人のために、選択のポイントや接続のノウハウなどを紹介する。——編集部, ASCII, 10月号, 233-240pp.

### ▶Map the Digital 電子地図への招待

地図の応用範囲を広げてくれるのがデジタルマップである。現在入手できる電子地図を紹介し、その魅力に迫る。——編集部, ASCII, 10月号, 329-335pp.

### ▶バカババのモノを買い物

「通販生活で米国グッズ紹介の巻」。バイザーにファンがついた帽子や、飲み干すと中からカエルが出現するマグカップなどおかしなものがいっぱい。——バカババ, ASCII, 10月号, 356-357pp.

### ▶パソコンにおけるマーフィーの法則

アスキーから出版された「マーフィーの法則」にちなんで、自分たちの身の周りにおけるマーフィーの法則を探す。——編集部, ASCII, 10月号, 382-383pp.

### ▶コンパクト・ディスク

CDの特徴や規格、現在ある種類を解説し、情報メディアとしてのCDのあり方を考える。——英斗恋, I/O, 10月号, 96-101pp.

### ▶RS-232Cクロス・ケーブルの製作

メディアサイズの違うパソコン同士で簡単にデータを共有するなら、RS-232Cを使うのもひとつの手である。そのケーブルを自作してみようという試み。——シャーマン, I/O, 10月号, 104-105pp.

### ▶赤外線レビータの製作

赤外線リモコンの信号を受信して、遠くへ転送してくれるレビータの製作だ。——和田好司, I/O, 10月号, 106-109pp.

### ▶スーパーコンピューティング入門

今回は自然現象のフラクタルを整理し、物理化学現象の例として「凝集」のシミュレーションを扱う。——林智雄, I/O, 10月号, 140-141pp.

### ▶ビジネスマンのための情報管理術

シャープのハイパー電子システム手帳用のICカード「ハイパー関数プログラムカード」を紹介。プログラムの書き込み実行例も挙げられている。——塚田洋一, My Computer Magazine, 10月号, 170-173pp.

## X1/turbo/Z

### X1シリーズ

#### ▶JEWEL BOX

コラムス風の宝石を使ったりパージゲーム。2プレイヤー対戦専用。——Electronics Forest, マイコンBASIC Magazine, 10月号, 138-139pp.

### X1turboシリーズ

#### ▶「お餅屋さん」のアルバイトだげーむ

11台のコンロの火をうまく調整し、いかにたくさんのお餅をきれいに焼きあげるかを競うゲーム。——HELL, マイコンBASIC Magazine, 10月号, 140-141pp.

## X68000

### ▶最新ゲーム徹底解剖!!

最新ゲームを長期にわたって攻略するページ。X68000用では「項劉記」「信長の野望・覇王伝」を取り上げている。——編集部, LOGIN, 18号, 106-143pp.

### ▶X68030新聞

新作盛りだくさんで、「コットン」「ネメシス'90改」「クレイジーライマー/クレイジーライマー2」「スーパーリアル麻雀PII & PIII」といったビッグタイトルを一挙



紹介。——編集部, LOGIN, 18号, 210-211pp.

#### ▶最新ゲーム徹底解剖!!

X68000用「ロボットコンストラクションR.C.」「項劉記」「信長の野望・霸王伝」など最新ゲームの攻略法。——編集部, LOGIN, 19号, 110-151pp.

#### ▶懐かしのGAME REVIEW

発売後、ずいぶんたった懐かしのゲームを紹介するコーナー。「サンダーフォースII」「三國志」が取り上げられている。——編集部, LOGIN, 19号, 152-153pp.

#### ▶X68030新聞

「ネメシス'90改」の特集。「グラディウス」シリーズの流れを対談形式でおさらいする。——編集部, LOGIN, 19号, 224-225pp.

#### ▶Dengekiパソコン

新作ソフトの紹介コーナー。X68000用は「コットン」ほか2本が取り上げられている。——編集部, 電撃王, 10月号, 73-83pp.

#### ▶SUPER SOFT EXPRESS

各機種用最新ソフトの内容を伝える。X68000用は「コットン」「ネメシス'90改」「クレイジークライマー/クレイジークライマー2」「レSSLエンジェルス」が登場。——編集部, コンプティーク, 10月号, 53-73pp.

#### ▶How to Win

すでに発売になったゲームを解説を交えて攻略する。X68000用には「項劉記」「信長の野望・霸王伝」「大航海時代II」の3本。——編集部, コンプティーク, 10月号, 79-123pp.

#### ▶NEW GAME REPO!!

各社の新作ゲームソフトの内容を紹介。X68000用は「クレイジークライマー/クレイジークライマー2」「ロボットコンストラクションR.C.」が登場。——編集部, テクノポリス, 10月号, 14-41pp.

#### ▶COMING SOON!!

SPSから発売される「ネメシス'90改」やブラザー工業の「宝魔ハンターライム第4話」など、未発売のゲームの最新情報。機種別カレンダーもある。——編集部, テクノポリス, 10月号, 42-57pp.

#### ▶HOT REVIEW!!

発売されたゲームを有名人にプレイしてもらい、その感想と環境を掲載する。千之ナイフ氏の「項劉記」ほか。——編集部, テクノポリス, 10月号, 64-81pp.

#### ▶DO-JIN SOFT FAN!!

先日行われたコミックマーケット44に出展された新作ソフトをまとめて公開する。X68000用ソフトも多数出展。——編集部, テクノポリス, 10月号, 86-99pp.

#### ▶まるごとD.O.美少女ディスク

美少女ソフトメーカーD.O.のグラフィックを付録ディスクに収録。——編集部, POPCOM, 10月号, 137-138pp. (ディスクの使い方)

#### ▶竜の巣

勇者を操作し、襲いかかってくる竜の頭を狙って倒せ! ——長瀬大学, マイコンBASIC Magazine, 10月号, 142-144pp.

#### ▶モンスターブリーダー

好きなモンスターを選んでトレーニングし、「悪の魔王主催 世界統一最強モンスター決定戦」を制覇する。——間部靖史, マイコンBASIC Magazine, 10月号, 145-147pp.

#### ▶STARBLADE

X68000+NAGDRV+GS音源用ミュージックプログラム。ナムコの「STARBLADE」より「The Theme of STARBLADE」。——加賀和孝, マイコンBASIC Magazine, 10月号, 159-161pp.

#### ▶SUPERSOFT HOT INFORMATION

電波新聞社「The World of X68000」ほか発売予定のX68000用ゲームを紹介する。——編集部, マイコンBASIC Magazine, 10月号, 別冊10p.

#### ▶ネメシス'90進化論

年末発売予定の「ネメシス'90改」について詳しい内容と注目点を伝える。——佐久間亮介, マイコンBASIC Magazine, 10月号, 188-191pp.

#### ▶FREE SOFTWARE INDEX

主要ネットにアップロードされたソフトウェアのなかから注目すべきものを紹介するコーナー。X68000用キー

ボードカスタマイズソフト「keyset.x」など。——編集部, ASCII, 10月号, 391-399pp.

#### ▶なんでもQ&A

「Human68k ver3.0ではCONFIG.SYSを書き換えられる機能があるそうだが?」などの質問に答える。——シャープAVCシステム事業推進室, My Computer Magazine, 10月号, 190-191pp.

#### ▶HOBBY EXPRESS

コナミの「悪魔城ドラキュラ」を紹介。何度もいろんな機種向けに作り直し、最新の技術を実践にフィードバックする姿勢を評価している。——あゆさわかすみ, My Computer Magazine, 10月号, 208-209pp.

#### ▶GCCで学ぶX68ゲームプログラミング

ゲームプログラミングの総仕上げ。最後に、もう一度プログラミングの要点をおさらいしてしめくくる。——吉野智興, C Magazine, 10月号, 159-163pp.

## ポケコン

#### PC-E500

#### ▶GUNMAN SPIRITS

1対1の早撃ち対決ゲーム。8人の早撃ちの名手たちを倒すのだ。——榎原隆史, マイコンBASIC Magazine, 10月号, 149p.

## 新刊書案内



ゲーム・オーバー  
任天堂帝国を築いた男たち  
デヴィッド・シェフ著  
角川書店刊  
☎03(3817)8521  
四六判 433ページ  
2,300円(税込)

私が任天堂に興味を持ったのは、とあるアメリカのパソコン関係者のパネルディスカッションにおいて、やたら「ニンテンドー」という言葉が出て来たときである。彼らは、当時の日本人以上にパソコンの将来を考えたとき、任天堂を一番の脅威と見ていたのだ、それはおそらく正しかった。

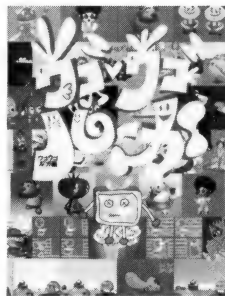
そんな任天堂の本がアメリカで出版された。お馴染みの企業もののビジネスノンフィクションとして、その翻訳が「ゲーム・オーバー」である。いきなり京都の任天堂創立時代から話は始まると妙な違和感を感じるが、それは些細なこと。アメリカではいきなり登場したNOA(ニンテンドーオ

ブアメリカ)の脅威のビジネスストーリー、アタリ社が作り上げ、そして廃墟にしたビデオゲーム市場を突如と復活させ、アメリカ中を席巻した驚異のストーリーが展開するのだ。

本書を読むと、ニンテンドーがいかにアメリカ中を巻き込み、シビアで独占的な手法を駆使して君臨してきたかがよくわかる。自社の利益とシェアを追及するシビアさ、ソフトウェアの重要性への先見、ソフトウェアの質を保つための強引な戦略と自社開発、3代目社長のカリスマ性。マイクロソフトやアップルなどのサクセスストーリーとは似ているようでどこか違うのがまた面白い。かなり辛辣な書き方もしているが、決してジャパンバッシングな本ではない。さわめて冷静に描かれている。

アメリカ社会で訴訟などのさまざまな軋轢に打ち勝って急成長した企業の物語としても面白いし、任天堂について深く調べることで、家庭用エンターテインメントマシンがどういった人々によって作られ、これからどうなるかとしているかを読むのにもいい。とにかくにも目を惹く点は多く、ゲーム一読すべきだ。

残念なのは、日本企業の話でさえ、翻訳ものに頼らねばならない日本の現状か。(K)



ウゴウゴルーガ  
ウゴウゴ・ルーガ編  
フジテレビ出版刊  
レブリッジセンター発売  
☎0424(88)9503  
A5変形判 126ページ  
1,750円(税込)

ついにというかやっと登場したウゴウゴルーガの本である。いきなり「おきらくごらく、おきらくごらく」とテレビくんがしゃべるのだ。そのほかにもテレビでお馴染みのあのユニークなキャラクターたちが、本のなかをところせましと埋めつくしている。まさにテレビで放送されたシーンのままに、それぞれのイメージを壊さないよううまく紙面に再現されている。

これはウゴウゴルーガのテレビ放送を見ていない人にはちょっと楽しめないかもしれないが、ウゴウゴルーガを朝見ないと目が覚めないというフリークは必ず持っていたい1冊である。



本とコンピュータ  
津野海太郎著  
晶文社刊  
☎03(3255)4501  
四六判 280ページ  
2,300円(税込)

あらゆるところでコンピュータが利用されるようになって久しい。それは出版業界においても同じであった。そして、電子出版の出現によって紙の本がなくなるかもしれないと、まるで「本」と「コンピュータ」は敵対関係にある2つの文化のように感じてしまう傾向も生まれてきている。

著者はコンピュータの文化に見られる、本の文化からの連続性や両者の共通性について語っている。そして「パーソナルコンピュータはたんなる道具ではない」という仮説が生まれてきた1970年前後のアメリカの文化環境を中心に、本とコンピュータの関係を論じている。



LIVE in'93などでCM-64やSC-55の演奏をSC-55mkIIで完全に聞けるのでしょうか？ またPCM8.XはノーマルX68000だと動作させるのがつらいでしょうか。

滋賀県 水谷 国宏



まず、音源の特性や使用できるパーシャル数の問題などからCM-64の曲を再現させることはできません。MT-32互換モードやCM-32 P互換バンクといったものもありますが、これらが完全に動作したとしても、最低2台のSC-55mkIIが必要になります。CM-64のデータは再現できません。

さらにSC-55のデータをSC-55mkIIで演奏する際にはいくつかの問題点があります。

ひとつは旧GS規格の「キャピタル落ち」という動作の仕様がなくなっていることにあります。これは拡張された音色バンクの音を使用されているとき、再生する機種側がその音色バンクを持っていない場合にキャピタルに近い音で自動的に代用するという機能です。

SC-55mkII以上に音色バンクが拡張された機種というのはありませんので、本来ならば問題がないはずなのですが、実際には演奏データ側で間違った音色バンクを指定している場合に問題が発生しています。ゲーム音楽でSC-55mkIIでちゃんと再生されていないものはこれが原因です。

もうひとつは音源仕様の細かな違いに関するものです。SC-55とSC-55mkIIでは、使用されているPCMデータはほとんど同じ、音源仕様も一部を除いて拡張されているだけなので、SC-55用のデータではかなりの再現性を見せます。しかし、厳密な意味ではSC-55mkIIはSC-55の上位コンパチではありません。

同じ値を指定した場合、エフェクトや音色パラメータなどの効き具合がかなり強めになってしまうようです。なんの細工もない曲ではたいした差は感じられないのですが、それらをいじった「凝った曲」ほど違ったニュアンスで再生されてしまいます。

当然、これはSC-55mkIIで作られた曲に対しても逆の意味で同じことがいえます。

ローランドではGS規格を変更して、SC-33、SC-55mkIIなどの仕様にあったものを新たなGS音源として再度標準化を進めていくことにしたようです。よって今後は、

従来のSC-55、SC-155、CM-300、CM-500、JV-30などはGS規格外の製品ということになります。

どちらもGM規格には適合しているのですが、GM規格という枠ではかなり妥協しないとデータが共用できないので、GM音源用といっても厳密なところでは機種限定しておくのが無難といえそうです。あるいは音源にほとんど依存しないか、ちょっとくらい違ってもかまわないようなデータ作りをするしかありません。

旧仕様のCM-300などはまだ販売されているようですし、すでに大量のHGS音源が出回っていることを考えると、現状ではまだまだ旧規格を標準と考えたほうがよさそうです。

次の質問ですが、PCM8を使った場合、演奏処理はかなり重くはなります。特に10MHzの機種では重さがかなり顕著に表れます。しかし、演奏自体にはよほどのことがないかぎり問題はありません。たとえばAD PCMを8チャンネル使っていたとしても、それがドラムだけであれば10MHzでも支障なく演奏することが可能です。

すでにPCM8を使った曲で10MHzでは正常に演奏できないというデータがありますが、これらはメロディラインにAD PCM音源を使ったデータなのです。メロディラインにAD PCMを使うことでFM音源では作れないようなリアルな音が出せるのですが、半面、こういったものは長時間AD PCMを占有するような構成になりやすく、結果として非常に重くなります。困ったことにこういったものは今後増えてくることが予想されます。

こういったものを10MHzで演奏することはできないのでしょうか？ ある程度までなら対処が可能です。

使用するデータをAD PCM形式ではなく、16ビットPCMのかたちで持つことによりPCM8の処理を軽減することが可能です。なお、16ビットPCMはZVT.X-Cで作成してください。あとはPCMトラックの頭で“@F5”を指定してください。



コマンドラインからスイッチなどを受け取ってそれをSTR型の変数に代入するX-BASICの外部関数を作り、コマンドシェルの外部コマンドをBASICで作りたいのです（C言語は避けて通りたいので）。引数は1(A2)から

00<sub>h</sub>が検出されるまでの内容とすればよいと思うのですが、その場合のパラメータIDはどうしたらよいのでしょうか。また、STR型の戻り値がほしい場合のパラメータIDを教えてください。東京都 小出 弘貴



確かに、プログラム起動直後のA2レジスタの内容から起動されたプログラムの情報を参照するというのはアセンブラでコマンドラインパラメータを得る際の常套手段です。しかし、コンパイルされたプログラムは途中で必ずBASICとC言語用の初期化関数を通りますので、A2にコマンドシェルからの情報が残っている可能性は低いといえるでしょう。

以下、質問の内容からは大きくはずれませんが、要するにX-BASICとコンパイラでコマンドライン上でパラメータつき実行可能なコマンドを作成するための方法を知りたいのだと判断して回答します。

このような処理に関する解説は1991年7月号の特集「パーソナルツール、BASIC」で行ったことがあるのですが、かなり簡単に済ませてしまっていたので、ここでもう一度詳しく解説しておきましょう。

BASICプログラムはコンパイルすることでC言語で作成したものと同様な実行形式を得ることができます。C言語ならコマンドラインパラメータを入手することは簡単です。そのためのインタフェイスが最初から用意されているからです。

BASICプログラムをコンパイルするとき、CC.Xは内部でBC.Xを呼び出してBASICのテキストファイルをC言語のテキストファイルにコンバートします。この段階で、実はコマンドラインパラメータを受け取るためのインタフェイスが付け加えられているのです。コンバートされたプログラムに見られる、

```
main(b_argc,b_argv[])
```

という部分がそれです。これをうまく使ってやればいいことになります。

具体的には、プログラム実行中にb\_argcという変数を参照すると、コマンドラインから指定されているパラメータの数がわかり、その内容はb\_argv(n)のようにすればn番目のコマンドラインパラメータとして参照できます。このとき、b\_argcはint型変数、b\_argv()はstr型配列として与えられます（BC.Xが勝手に加えてくれますので、b\_



argc, b\_argv()を宣言する必要はありません)。

注意としては、コマンドラインパラメータにはコマンド名自身も含まれるのでb\_argcは常に1以上になります。

たとえば、

A>TEST AAA.DAT /T100 /W

のようにしてTEST.Xというコマンドを実行した場合、

b\_argv(0)にコマンド名“TEST”

b\_argv(1)に“AAA.DAT”

b\_argv(2)に“/T100”

b\_argv(3)に“/W”

という文字列が格納されることになります。

b\_argcは4になっています。あとはこれをうまく解釈すれば処理は終わりです。

ちなみに、これだとコンパイルしなければ動かないプログラムになってしまいますが、コンパイルされているかどうかを判定することによって処理を分ければ、ひとつのプログラムでコンパイルしてあるかどうかにかかわらず動くものにもすることも不可能ではありません。

さて、こういったことをふまえて簡単な

サンプルを作ってみました。リストを見てください。

このサンプルで想定しているのは、コマンドラインパラメータで設定される要素が、

ファイル名2個

オプション/A~/Z

(それぞれに数値パラメータを持つことができる)

のようになっているものです。オプションの指定は“/”のみ有効になっていますが、“-”を許容するようにしたほうがよいでしょう。

ファイル名やオプションはどういう順番になっているても正しく指定されることが望ましいといえます。たまにオプションの位置はコマンドの直後だけに限定されたツールなどもありますが、柔軟性に欠けるので使い勝手はあまりよくありません。

そこで、ここでは指定されているものを順番に調べています。指定されているかどうかはmode()配列にフラグとして格納されています。1で引数つき、2で単に指定のみ、0なら未指定です。実際のプログラムではこんなにたくさんのオプションに対

応させる必要はないのでその場に適した処理を記述することができそうですが、汎用性を考えてこのような処理にしてみました。

大文字小文字の区別は行っていませんが、もちろん分けることも可能です。使い勝手を考えると大文字小文字の区別はしないほうがよいと思われますが……。

このサンプルでは、オプションに続いて指定できる値として整数の処理以外は考慮されていません(実数対応にするのは配列value()をfloatで宣言するだけです)。文字列型などを使いたいときは別途処理を組んだほうが話が早いでしょう。

ちなみに、コマンド名だけで起動されたとき(b\_argcが1)はヘルプ表示を行います。

注意としては、このような処理はプログラムのメイン部分に置いておかねばなりません。コマンドラインパラメータはmain()関数の引数として渡されるのでユーザー関数の中にまとめることはできないようです。また、コマンドラインから違和感なく使用するためにはBC.XでCに変換されたプログラム中から、

b\_init();

の行を削除し、

b\_exit(0);

の部分で、

\_exit(0);

のように変更しておく必要があります。

(中野 修一)

## リスト

```
10 /* main(b_argc,b_argv())
20 /*
30 str filename1="nul"
40 str filename2="nul"
50 int mode(25)
60 int value(25)
70 int ppp
80 /*
90 int i,j,k,l,m
100 int strn(9)
110 if l=1 then {
120   for i=1 to b_argc-1
130     if left$(b_argv(i),1)<>"/" then {
140       k=k+1
150       strn(k)=i
160     } else {
170       l=tolower(asc(mid$(b_argv(i),2,1)))-'a'
180       value(l)=val(mid$(b_argv(i),3,255))
190       if strlen(b_argv(i))>2 then mode(l)=1 else mode(l)=2
200     }
210   next
220   if k=1 then filename1=b_argv(strn(1))
230   if k=2 then filename2=b_argv(strn(2))
240 /*if k=N then filenameN=b_argv(strn(N))
250   ppp=b_argc
260 }
270 /*
280 if not ppp<=1 then {
290   print "ファイル名1は";filename1;"です"
300   print "ファイル名2は";filename2;"です"
310   print "オプションスイッチは"
320   for i=0 to 25
330     if mode(i)<>0 then {
340       print chr$( 'A'+i);" が有効です。値は";
350       if mode(i)=1 then {
360         print value(i);" です"
370       } else print "ありません"
380     }
390   next
400 } else {
410   print "ヘルプメッセージ:適当に指定すること"
420 }
430 end
```

## 質問にお答えします

日ごろ疑問に思っていること、どんなことでも結構です。どんどんお便りください。難問、奇問、編集室が総力を挙げてお答えいたします。ただし、お寄せいただいているものの中には、マニュアルを読めばすぐに解答が得られるようなものも多々あります。最低限、マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名、システム構成、必要なら図も入れてこと細かに書いてください。また、返信用切手同封の質問をよく受けますが、原則として、質問には本誌上でお答えすることになっていますのでご了承ください。なお、質問の内容について、直接問い合わせることもありますので電話番号も明記してください。

宛先: 〒103 東京都中央区日本橋浜町

3-42-3

ソフトバンク株式会社出版部

Oh!X編集部「Oh!X質問箱」係



Oh!X 1993.11.



◆ウッキー キキキ キー! (祝 善バビ! ページ化!) これ、エレガント押し花電報でお願いします。 横山 純一(18)東京都

料金はどちらに請求すればよろしいでしょうか。今回はサービスで口頭にて伝えておきますね。

◆缶飲料の製造日の話がありました、結構古いものでいけるのではないのでしょうか。昨年の夏に信州で「柿マスカット」「柿レモン」を買ったら1987年の製造でしたが、別になんともありませんでした。味はなんともなくはなかったのですが……。6年近く売れ残ったのもったもな気がしました。 神田 望(21)東京都

丈夫な胃がうらやましい。6年のあいだに味が変わった可能性はないのでしょうか。どんな味だったかも詳しく教えて。

◆夏コミに行ってきた友人が「ドクターペッパー」を買ってきてくれたのですが、1本しかないので飲まずにモニタの横に置いています。ううっ、昔は大阪でも売ってたのになあ……。 村上 剛規(20)大阪府

冬のおみやげもきまりですね。ところで、その「ドクターペッパー」、何年くらい寝かせる予定なんですか。

◆フロッピーディスクが意外にヤワな記憶媒体であることを知って、いまあわてております。実は5年ほど前のフロッピーディスクを読み込ませようとしてCRCエラーをくらってしまったんです。それも1枚だけならまだしも10枚中2枚死んでいるではありませんか。ちょっとシャレになりませんよね。やはり大事なデータはHDかMOに保存しないといけないのでしょうか?

渡辺 久孝(26)大阪府  
こまめにチェックすること、保存の状態が重要なのでは。ちなみに家にある4年前のフロッピーディスクは無事でしたよ。

◆最近、Oh!Xがつまらなく思える。周囲の68ユーザーでも購読をやめたのがいる。本よりパソコン通信のほうが情報が早いし、内容もあるのでOh!Xを読むメリットが減ったからだと思う。8年前のようにページをめくる楽しみがない。そこかしこに情報があふれているからだろうか。

大野 政治(26)大阪府  
たしかにパソコン通信のほうが自分にとって必要な情報を引き出しやすく、早いかもしれません。だからといって本が不要なわけではないと思いますよ。パソコン通信とは別のアプローチで読者に必要な情報を提供できればと考えています。

◆最近、アニメや特撮モノの主題歌のあるカラオケ屋を見つけまして友達と歌いまくっています。思いきり叫ぶ曲ばかり歌わされるのでノドがかわてしまいます。しかし、ストレス発散にはなってると思う今日このごろであります。

上田 考一(23)福岡県  
大声を出すっていうのは本当に気持ちいいですね。「歌わされる」なんていって本当はマイクを放さないんじゃないですか(ありがちなツッコミ)。でも毎回そんな曲を歌



それにしてもドラキュラは難しいよね。ちなみに、この女性に「誰かウチですばらしい演出を見せてください」と祈っているのかなあ。ちがう。



「橋本 和典 東京都 WILLOWをほしうにしているコットンがとけていいですね。ところで、予約特典のあの「湯のみ」を手に入れましたか? 私はううっ……」

っているなんて、とてもここには書けません(書かないともっと怪しい)。

◆うっ、Oh!Xにミロをこぼしてしまいました。なに、ざまーミロ? しっ、失礼しました(バカ)。ところで昔から友人Mに雑誌を貸すとチョコレートだのポテトチップスだの、ときにはカレー(ごはんつき)をはさんで返すので、ぼくはいつも「彼は雑誌で押し花でもしてるんだな」と思っています。あっ、押し花ではありませんね。 斎藤 仁(18)岐阜県

たまにはお金でもはさんでくれると嬉しいんですけどね。

◆Oh!Xをぬれたテーブルの上に置きます。そして1~2時間すると、Oh!Xがテーブルにくっきます。それから無理にとろうとすると、表紙がはがれます。みなさん、こんな経験ありませんか? 齋藤 真二(19)東京都  
みなさん、こんな経験ありますか? きっと本を大切にしてくれてるから不思議ですね。ないといつて。

◆最近、新しくヘッドホン(P社のRP-F30)を買ったのですが、これがものすごく音がいい。超低音から超高音までレンジは広いし、音の解像度がよくて今まで気づかなかった音まで聴こえます。音採りなどする際には最適なのではないのでしょうか。ヘッドホンのくせに「アフターバーナー」がついてるし(笑)。欠点は密閉型なので長くつけていると暑い、ということくらいでしょうか。私はこれより高いヘッドホンも持っているのですが、このごろはもっぱらこれで聴いています(ちなみに自作のヘッドホン用アンプを使用)。 植木 正幸(24)神奈川県  
やっぱりいいものに巡り合うと嬉しくなっていくついに周りに薦めたくてしまうもの。またなにか見つかったら教えてください。

◆C.G.イラストレーターのみなさん! 笠原弘子のビデオ「マインド・ギャラリー」(ワーナーミュージック・ジャパン発売)はいいですぞ。特に自然画を描きたい人にオススメです。なにがいいかというと家にいながらにして、湖、森林、草原などを一度に見られるんですから。ほかに洋式の家(?)もあるし、一見の価値あり。

江ヶ崎 貞行(20)千葉県

たまには心を落ち着けて、ゆっくりとこんなビデオを見るのもいいでしょうね。

◆このハガキを書いているとき家のネコがジャマしに来た。手をかんだり、ハガキの上ののったり。最近遊んでないもんねえ。でも、プリンタの上で寝るのはやめてね。

松永 司(18)大阪府

だからハガキの字がきたないんですね。今回は大目にみましょう。でも、プリンタの上で寝心地いいんでしょうか。プリンタが潰れるような気がするんですけど。

◆この前よその家のネコが勝手にひとの家にがり込んでウチのネコとけんかしたうえ、「しっこ」までしていきやがった。チクショー。

佐藤 友一郎(20)宮城県

そのけんか、もちろん勝ったんでしょね。なに、負けた? 今日から特訓、ミッキー・ロークに弟子入りですね(古いなあ)。

◆9月号の129ページを見て思いました。読者はもっとOh!Xに貢献しなければなあ。アンケートはもうやるから次のステップはプログラムがイラストでしょう。いずれはスタッフとして、ふっふっふっ……。 柄多 英樹(21)北海道  
お待ちしてますからね。忘れませんよ、絶対に。おいおい脅してどうする。みなさん、アンケートハガキ、いつもありがとうございます。今月号もよろしくね。

◆Oh!Xのせいで、買っている雑誌では1度でいいからプレゼントに当たるという野望が果たせないでいるんですよ。 小山 優一(19)東京都  
そんなこといってると、後ろから誰かに襲われますよ。世の中にはプレゼントに1度も当たったことのない人も多いんですから。

◆プレゼント当選の送付案内書って、直子の代筆で書いてませんか? 大野 敏郎(18)岡山県  
そんなことはありません。某国民機「一〇郎」なんてことなく、付属のワープロ「WP.X」で書いてますよ。みなさんはいっこれを見られるでしょうか。

◆STUDIO Xで、姉弟4コマを描いていることは姉には内緒なのです。ですので、結婚式の日にお祝い電報などはご遠慮ください(笑)。いや、

まじでまじで。岡村 直也(22)兵庫県  
お姉さんは知らなかったんですね。フィクションとはいえ、自分をモデルとしたあのキャラクターを知らないとは。では、Oh!Xを1冊、お送りしないといけなんでしょうか。

◆小じわの増えたあゆはらこずえちゃんが回転レシーブをしている姿には驚いてしまった。

平田 恭敏(18)埼玉県  
あのお菓子のCMに登場するやつですよ。初めて見たときはビックリして声を上げてしまいました。商品名はど忘れしてしまっただけ、結構インパクトがあったような気がします。

◆20km/hって速いんだなあ(初めて運転した男)。  
小林 敦(18)埼玉県  
5kmを15分、時速20km。マラソンランナーって本当に速いですよねえ。えっ、人の走る速度じゃないんですか。

◆広告をもっとたくさん入れればOh!Xの値段も安くなるのでしょうか? 保坂 哲広(23)愛知県  
基本的にはそうです。個人の広告でもみなさんいかが?(冗談ですよ)。

◆ようやく就職が決まりました。親を捨て、ひとりで……いや、愛しのX68000ちゃんと共に東京へ旅立ちます。やったあ、4月から東京人です。これで編集部が近くなりましたよ。

北本 信幸(20)石川県  
4月までまだまだありますが、東京にてお待ちしております。残りの学生時代を楽しんでくださいな。

◆就職が決まりました。いま大人気の公務員です。札幌市役所。もともと第1志望だったので、合格通知が来たときは、とても嬉しかったです。世の中、不況だなんだと騒いでおりますが、真面目にコツコツやってきた人間がバカを見ることのない世の中になったような気もしております。波に乗っていると、逆風のときに困るんですよ!

新井 誠治(22)北海道  
これからも一生懸命書いていってくださいな。疲れたときの休憩のお供にはOh!Xを忘れずに。

◆ボランティア活動に参加したときに、いっし

ょに活動していた女の子にホレてしまいました。今度、人生最大の賭けに挑むつもりです。「恋愛雑誌Oh!X」のお力があれば、きょうまくいくでしょう。P.S. 武田さんのイラスト大好きです。

坂井 国彦(21)静岡県  
恋愛も請け負っていたのか、忙しいなあ。どの程度の力があるかわかりませんが、陰ながら応援させていただきます。

◆9月号146ページの富田さんへ。私は仮免を取って初めて路上教習に出た日、犬をひいてしまいました。犬の場合は無視して通り過ぎてよいのだそうです(教官談)。加藤 隆(22)佐賀県  
いくら無視していいとはいっても、やっぱり気になるのが人情ってやつじゃないですか。話し方が変です。ちょっと怒ってるかな、誰に対してというわけではないんですけど。

◆あと150円の林寛さんはアンケートハガキにちゃんと「リンカーン」ってフリガナを振っているのでしょうか。そういえば過去に「サントクロース」(黒須三太)さんもありましたね。ほかにたったの20日で悟りの境地を開く(廿日出悟)釈迦のような人や勧善懲惡「セーブマン」もいますね。Oh!Xについていろんな人が読んでいるんですね。いやはや。坪田 雅己(18)広島県  
とっても楽しいんですけど正しいフリガナも振っていただきたいね。

◆最近なぜお金がない。給料日がきても翌日にはなくなってる。X68030がなぜか遠のく。結婚ってお金がかかるものなのね。でも彼女のためなら……。谷口 博一(27)大阪府  
ごちそうさまです。でも結婚してもいないのにお金がないのはなぜだろう。

◆今度結婚します。彼女のほうは、ワープロは使いますがパソコンはわからないようです。アクションゲームはつらそうなので、パズルゲーム(上海など)やアドベンチャーゲーム(私と同じ趣味でミステリー好き)などをやらせてみようかと思っています。横堀 正敏(29)埼玉県  
幸せそうでいいですね。風が心に冷たい季節だというのに。そういえば、アクションゲームでもリブルラブルなんかはいかがでしょうか。

◆我が愛車のカラーIIは成仏してしまいました。先日、突然エンジンの回転数が上がりだし、水温計がレッドゾーンを越え、針が振り切ってしまいました。近くの修理屋に突っ込み、エンジンをばらしたところ、シリンダー内壁に深さ1~1.5mmほどの傷が……。エンジンを総とっかえししなければならないとのこと。私は購入価格25万(来月車検)の愛車を手放し、次の車を探すのでした。寺門 修司(22)兵庫県

そんな、傷つけるなんてひどい! 不可抗力ですって。いつもそんなことばかりいって。もういや。そして彼女(車)は去ってしまいました。なんてね。

◆9月号の村瀬さんは甘い! 私は20歳になると、「大人の仲間入り」よりも「もう10代じゃなくなる」という気持ちのほうが強く、誕生日は落ち込んでいました。そのため、直後のお正月もぜんぜんおめでたくありませんでしたよ(誕生日は12月30日)。奥原 祥(20)北海道  
10代に対する思い入れが強いんですね。10代でなくなることで、なにが大きく変わるのでしょうか。

◆これから沖縄へ行ってきます。そんでもって初めて飛行機に乗るんですよ。行く前からもうウキウキワクワクです。ところで、編集部のみさんの夏休みってどれくらいあるのかなあ。

白木 剛二(20)愛知県  
なにをいまさら夏休みなんて。冬こそは遊ぶぞー。でも、その前に年末が……。

◆東京ディズニーランドに初めて行きました。お盆の真ん中だったので激混みでしたが、非現実的ななかにとっぷりつかれて、よかった。ただ、13時間30分のうち、並んでいた時間が10時間近かった。

嶋 真一(28)大阪府  
「非現実的ななかで、とっぷり疲れてよかった」ですか。やっぱり10時間も並ぶなんて非現実的ですからね。えっ、そうじゃないんですか、失礼しましたー。

◆MOって「もー」って読むんですよね? えっ違う。

松尾 美千代(26)大阪府  
えっ、そう読むんじゃないんですか? 後ろから「エムオー」だって声が聞こえてきます。「みんな知ってる」だそうです。

◆ホッホッホッ……私が隠れ高齢者です。ところでX68000のグラフィック機能に魅せられてユーザーになった人たち(私も含めて)はINDYや3DOは惹かれるところがあるでしょうネ! ちなみに私はシリコングラフィックスのPartners Directoryを入手して調べましたが、アプリの高いこと、1桁~2桁金額が違います。代理店の営業マンは、メイン32Mバイト、HD1Gバイト、24ビットプレーン、17インチモニタのシステムを推奨してくださいました。基本システムだけでもいいからほしい! これでD6GAが走れば最高!

加藤 治(47)三重県  
基本システムとはいえ、薄給の我が身にはつらいですよ。でもいつかは……。

◆このあいだ、東海道本線車中にて50歳代の男性がOh!Xとおぼしき雑誌を読んでいる場面に遭



▲紀内 亮 東京都  
発売の決定した「メメシス'90改」が早くも登場。みんなの期待もとっても高く、やっぱりって感じ。とってもムズイので腕を磨いておきましょう。



▲藤澤 篤 奈良県  
シモンの困っている顔がかわいいのね。でも、ニシモンかわいい女の子だったら、少女のことは許しなかわい女の子だったら、少々のはしゃべって。ちやうぞ。もっと、しゃべってしゃべって。



遇! オー、俺より年上のX68000ファンが実在したぞー、と喜んだ。声をかけて話し込めばよかった(反省)。

高橋 茂(42)静岡県  
やはり幅広い年齢層の方が読まれているようです。来月のハガキがいまから楽しみ楽しみ。

◆8月18日、編集部よりシャーペンとお手紙が届く。Oh!Xを買う。……載ってない。「やるな」とわけのわからんことをつぶやいてしまった。やっぱり、採用の基準がわからない。謎だ。それにこのシャーペンはいったいどの投稿作品に對するものなのかわからない。「採用が決定し

ました際には、別途ご連絡いたします」。本当ですね。ウソじゃないですね。いやあ、大変そうだなこれは。でも、なんでいきなり?……わからん……やるな。

佐田 匠(18)千葉県  
どうもご迷惑をおかけしました。多くの方がいきなり送られてきたシャーペンはなんだと思われたことでしょう。これは、イラストを除くプログラムなどの投稿に對してお送りしている記念品です。今回の記念品は製作が遅れ、半年ほど送るのがとどこおっていました。遅れたお詫びと共に、新たな投稿をお待ちしております。



▲江副 滋 埼玉県  
いまの自分の心境というのですが、すっきりとした表情ですね。私もこんな空なら見上げていた。そのときだけでも気が晴れそうですから。

## ぼくらの掲示板

- 掲載ご希望の方は、官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
- ソフトの売買、交換については、いっさい掲載できません。
- 取り引きについては当編集部では責任を負いかねます。
- 応募者多数の場合、掲載できないこともあります。
- 紹介を希望されるサークルは必ず会誌の見本を送ってください。

### 仲間

★MZ-2500のディスクマガジン「★DUST BOX」を発行してきた「星くずばこ」作成チームですが、今後のさらなる発展を期して「会員制」を採用することになりました。「MZ-2500を骨までしゃぶる」をポリシーに、ますます充実した活動を行っていきますので、MZ-2500/2800ユーザーの方、ぜひ参加をお願いします。求めるものがきつとある……かも。62円切手同封にて(見本ディスク希望の場合は72円切手6枚を追加)下記までご連絡ください。折り返し入会案内と申込書をお送りいたします。〒807 福岡県北九州市八幡西区星和町24-54 山ノ内方 星くずばこ

★「ICPソフト」では、同人ソフト開発メンバー(特にグラフィック関係)を大募集しています。詳しいことは、こちらから資料をお送りしますので、住所、氏名、年齢を明記のうえ、下記までご連絡ください。〒565 大阪府吹田市山田西2-4 AI-408 米村 貴裕(19)

### 売ります

★カラスキャナ「CN-8NSI」一式を50,000円で売ります。連絡は往復ハガキをお願いします。〒173 東京都板橋区本町31-1 武藤方2F 池田健一(26)

★シャープ製カラー熱転写プリンタ「CZ-8PC5-BK」、アイ・オー・データ機器製増設RAMボード「PIO-6BE4-4M」、それぞれ35,000円位で売ります。高く買ってください方優先です。連絡は官製ハガキをお願いします。〒960 福島県福島市蓬萊町43-24 金子 雅宣(29)

★Roland製MIDIモジュール「CM-32L」を30,000円前後で売ります。箱、マニュアル、付属品はあります。希望価格を書いて官製ハガキで連絡してください。〒939-16 富山県西砺波郡福光町新町41 松村 直樹(17)

★XI用FM音源ボード「CZ-8BSI」を8,000円、XI用カラーイメージボード「CZ-8BVI」を3,000円で売ります。どちらも送料込みで、箱、付属品、マニュアル有ります。〒028 岩手県久慈市中町1-38 熊谷 武志(24)

### 買います

★X68000用拡張I/Oボックス「CZ-6EBI」(グレー)を55,000円程度で買います。連絡は往復ハガキをお願いします。〒573 大阪府枚方市星ヶ丘2-25-10 森 秀樹(23)

★X68000用ハンディスキャナ「HGS-68」を15,000円程度で買います。完動品で付属品つきなら箱、説明書はなくても可。連絡は官製ハガキをお願いします。〒300-27 茨城県結城郡石下町大房873-1 大久保 典之(18)

★MIDIボード「CZ-6BM1」を送料込み10,000~13,000円で買います。説明書があると嬉しいです。安価優先。連絡は往復ハガキをお願いします。〒040 北海道函館市宇賀浦町5-24 太田 志輝(16)

★X68000用拡張I/Oボックス「CZ-6EBI」を45,000円程度で買います。色はグレー、黒を問いません。連絡は官製ハガキにてお願いします。〒432 静岡県浜松市和地山2-29-14 竹内正樹(20)

★X68000CompactXVI用2Mバイト増設RAMボード「CZ-6BE2D」を20,000円程度で、同5インチFDD(シャープが満開製作所)を30,000円程度、同プリンタケーブルを2,000円で買います。連絡は往復ハガキをお願いします。〒960-12 福島県福島市松川町字中町10 佐藤 雅哉(20)

★X68000XVI用の2Mバイト増設RAMボード「CZ-6BE2A」を25,000円前後で買います。完動品で付属品、説明書つき。連絡は官製ハガキをお願いします。〒425 静岡県焼津市焼市4-12-5 鈴木 善男(18)

★21インチカラーディスプレイ「CU-21HD」(黒)を50,000円前後で、RGBシステムチューナー「CZ-6TU-BK」(黒)がセットなら10,000円プラスして60,000円前後で買います。完動品に限りです。連絡は官製ハガキをお願いします。〒840 佐賀県佐賀市与賀町4-5 コーポ栄城5号 福知健(22)

★アイ・オー・データ機器製X68000用4Mバイト増設RAMボード「PIO-6BE4-4ME」を30,000円程度で買います。希望の値段と状態を書いて往復ハガキにてご連絡くださるようお願いします。〒230 神奈川県横浜市長見区朝日町2-89-312 村中 隆志(23)

★X68000XVI用の2Mバイト増設RAM「CZ-6BE2B」と数値演算プロセッサ「CZ-6BP2」をそれぞれ20,000円前後で買います。完動品で付属品、説明書つきのものを。箱なし可。連絡は程度、希望価格などを書いて往復ハガキをお願いします。〒065 北海道札幌市東区伏古9条5-4-24 穴田 裕之(28)

★X68000用MIDIボード(メーカー不問)+Roland製MIDI音源モジュール「SC-55」を40,000円で、「CM-500」ならば55,000円で買います。完動品で付属品、取扱説明書があれば、少々の傷は可。トラブル防止のため手渡し(関西圏)希望。連絡は官製ハガキをお願いします。〒630 奈良県奈良市大寺西1-288-37 寺本 賢二(20)

### バックナンバー

★Oh!X1986年7月号、1992年11月号をそれぞれ1,000円程度で買います。また、「ナムコ・ビデオゲーム・ミュージック・ライブラリーVol.1 For X68000」(機田健一郎著)を10,000円以上で買います。付属品、説明書などがあれば必ずつけてください。連絡は官製ハガキをお願いします。〒321 栃木県宇都宮市岩曽町1462-50 古橋 康弘(18)

## DRIVE ON

このコーナーでは、本誌年間モニタの方々のご意見を紹介しています。今月は9月号の内容に関するレポートです。

●9月号の特集記事の「もうMOしかない!」を読んで、ちょっとだけMOを買おうかと思ってしまいました。やはり大容量メディアとしてハードディスクの次はMOだと思いますし、SX-WINDOWで動画を扱えるようになったのだから、画像データとして1枚使うこともできます。そこで、ふと頭に浮かぶのが「つながるかな?」ということです。特集記事では、ほとんど問題なくつながるようですし、Human68kもver.3.0にすればいいようなので、ほっとしました。MOがかなり将来性のあるメディアだということが、改めてわかった気がします。

森崎 剛(21) X68000 XVI 広島県

●製品と中身のドライブの紹介が詳しく載っているの、これからMOを買おうという人には、非常に役立つ特集ではないでしょうか。特に新製品紹介の背面の写真は、とても参考になります。これなら安心してMOを導入できます(もちろんお金があればね)。しかし、MOにメンテナンスの必要があるなんて知らなかった。

石田 伯仁(20) X68030, MZ-73I, PC-8801 mkIIIMR, PC-E200 神奈川県

●9月号の特集は、MOの導入を検討している人にとって参考になったと思います。個人的には、ロジテックの製品に魅力を感じました。唯一のX68000対応ですし、メンテ用品同梱というところも好感がもてます。また、MOではありませんが、同社製品のハードディスクではパワースイッチが前面にレイアウトされており(本来なら当然?), やはり好感がもてます。ぜひともがんばってほしいメーカーですね。

橋本 和典(26) X68000 XVI, PC-9801RX2, Macintosh LC520 東京都

●うへむ、3DO。こいつは思っていた以上にとんでもないものかもしれません。しかし、なんでしょうね。36チャンネルのDMACって。これならフルアニメーションができて当然って気がします。しかし、ハードウェアでこれだけのことができて、しよせんは入力に対して用意された答えを返すものであることに、変わりないと思います(多少の柔軟性はあるとしても)。とりあえず、CD-ROMの容量という問題を抜きにして、ゲーム内での自由度の限界を感じさせないソフトを作ることが、現在のソフトメーカーにできるのでしょうか。でなければ、この強力な表示能力は、まさに宝の持ち腐れ。このすごいマシンが、絵のき

れいだけのつまらないゲーム機にならないことを祈っています。

吉岡 洋明(20) X68000 PRO II, PC-8801MA, FM NEW7 埼玉県

●9月号「こちらシステムX探偵事務所」の柴田さんのモーフィング実験は、なかなか面白そうですね。完成するのが楽しみです(実験に使ったのは現在の柴田さんの写真ですか)。ターミネーター2以降、テレビのあちこちでモーフィング映像を見ますが、質が非常に高いですね。

ぜひ、動画対応にしてX680x0でラモスをモーフィングさせましょう。

松永 孝治(23) Xturbo model30, MZ-80C, PC-9801N, AMIGA1200/85MB 鳥取県

●「ツインマウスドライバTMD.X」のコネクタが2つついているのだから、同時に使おうという発想がよかった。実際、マウスを使ったゲームで、イコールコンディションで対戦できるというのは楽しいでしょう(もちろんゲーム自体が面白ければ)。ただ、問題はマウスを2つ同時に使えるようなスペースがあるかでしょう。まあ、トラックボールで使えば問題ないのかもしれませんが。個人的には、片方のマウスで機体进行操作し、もう片方のマウスで照準を操るようなフライトシミュレータ的なゲームをやりたいですね。

北風 保(22) X68000 ACE 東京都

## ごめんなさいのコーナー

10月号 秋祭りPRO-68K

●ディスク4のゲームが動かない

・West Cliffの起動方法

ゲームの起動前にカードドライバを、  
CARD DRV TR.DAT  
のように組み込む必要があります。無事、組み込み終わったら、

West\_Cliff  
として実行させてください。

・CHERRY BOY

ZMUSIC.Xの組み込みが行われていないと、エラーを起こして正しく起動できません。

ディスク4に収録されているZMUSIC.Xを、  
ZMUSIC--SLOT\_SND.ZMS  
として組み込み、  
SLOT

で実行してください。起動バッチファイルCHERRY.BATを使う場合は、

CD ZM

COPY ZMUSIC.X B:¥CHERRY¥

のようにして、CHERRY BOYと同じディレクトリにZMUSIC.Xを転送してから、バッチファイルを実行してください。

・PENJANG!

起動バッチファイル名と実行プログラム名が同じであるため、起動バッチファイルを使う場合は、

PENJANG.BAT

と拡張子まで入力するか、起動バッチファイル名をリネームしてください。

音楽が正常に演奏されない場合は、音色データが正しく組み込めていない可能性があります。なお、基本的にZMUSIC.Xを使用するときには、OPMDRV.Xを外してください。

●SAVESYS.SYSが見つからない

これは、ディスク4の中のDRACURACLOCKと同じディレクトリに格納されています。

このほかにも、動かないなどの問題がありましたら、アンケートハガキなどに詳しい状況を書いて、Oh!X編集部までお送りください。随時サポートしていきたいと思っています。一部、説明不足などがありましたことをお詫びいたします。

10月号 Oh!X LIVE in'93

P.80 PASSING BREEZEを演奏させるためには、スーパーハングオンのPCMデータが必要になります。1993年8月号113ページに掲載されている「スーパーハングオンのAD PCMデータ分離プログラム」を使用して、AD PCMファイルを作成してから演奏をしてください。記事中に説明されていなかったことをお詫びします。

バグに関するお問い合わせは  
☎03(5642)8182(直通)  
月~金曜日16:00~18:00

お問い合わせは原則として、本誌のバグ情報のみに限らせていただきます。入力法、操作方法などはマニュアルをよくお読みください。また、よくアドベンチャーゲームの解答を求めるお電話をいただきますが、本誌ではいっさいお答えできません。ご了承ください。



## もう使った？ ううん、まだ おっくれってる～

▼ポリゴナイザライブラリ「SLASH」をどのようにして活用していくか。サポートツールの作成、基礎事項の確認をとおして、「SLASH」とはどういったものか、解説してみました。

現状では、まだ「SLASH」のシステムが固定化されておらず、使うことに不安を覚えるかもしれません。しかし、基本的なコンセプトはしっかりしているのですから、興味がわいたらとてあえず使ってみましょう。

もちろん、「SLASH」は生まれたばかりのシステムですから、システム以外ほとんどなにもありません。よりよい環境を構築するためにも、優秀なサポートツールが切望されています。ちょっとしたツールなら……という人は、ぜひ挑戦してください。

また、同じようなシステムを制作していた読者の方もいらっしゃるようですが、「SLASH」が発表されたからといってあきら

めず、ぜひ、自分の目標に向かってがんばってください。横内氏もいっていますが、「SLASH」がベストのシステムではありません。「SLASH」を超えるものを作って、横内氏を見返してやろうではありませんか。

▼いよいよ来月号でOh!X改題6周年を迎えます。あいかわらず、年2回の創刊記念号ということとでなんかやるんじゃないかな。と思っているんですけど……どうでしょうか。担当者をついても教えてください。

また、予告を見てもらえばわかるでしょうが、12月号はゲーム特集です。最近、あまり元気がないX68000のゲームですが、そんな雰囲気吹き飛ばすように元一杯ゲームを楽しめる特集にする予定です。

▼「X68000マシン語プログラミング」は著者スランプのため、「吾輩はX68000である」は著者急病のため、「知能機械概論」は著者多忙のため、今月はお休みとさせていただきます。また、「ハードウェア工作」は、三沢氏の都合がつかず、12月号も休載の予定です。毎月楽しみにしていただいている読者の皆さんには、申し訳ありませんでした。

### 投稿応募要領

- 原稿には、住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。
- プログラムを投稿される方は、詳しい内容の説明、利用法、できればフローチャート、変数表、メモリマップ（マシン語の場合）に、参考文献を明記し、プログラムをセーブしたテープ（ディスケット）を添えてお送りください。また、掲載にあたっては、編集上の都合により加筆修正させていただくことがありますのでご了承ください。
- ハードの製作などを投稿される方は、詳しい内容の説明のほか回路図、部品表、できれば実体配線図も添えてください。編集室で検討のうえ、製作したハードが必要な場合はご連絡いたします。
- 投稿者のモラルとして、他誌との二重投稿、他機種用プログラムを単に移植したものは固くお断りいたします。

あて先

〒103 東京都中央区日本橋浜町3-42-3

ソフトバンク出版部

Oh!X「㊟㊟㊟」係

## S H I F T ・ B R E A K

▶X68000とHDとテレビ。PC-9801とディスプレイ。X1turboとテレビ。SC-55とD-70とモニタスピーカ。あとはスタンドにテレビにコードレスホンにディスクマンに金鳥リキッドに扇風機。普通の暮らしをしているのにコンセントが足りない。タップは見た目が悪いからイヤなだけだな。電器業界の不振は、案外お部屋のインフラの問題かもしれない。(E.K)

▶家に帰ると留守電に用件が入っている。ほとんどが無言ののだが、録音されている時刻を聞くと、正午とかオヤツの時間のまっ昼間ばかり。偉そうなことはいえないが、まっとうな人間がこんな時間に家にいると思っているのだろうか。どうせなかのセールスだろうが、もうちょっと頭使わないと売れるものも売れないと思うぞ、うんうん。(八)

▶ジョージ・ルーカス展に行ってきた。映画「STAR WARS」3部作、「INDIANA JONES」3部作その他で使われた模型や合成に用いられたマットペインティングまで、400点も展示されていた。あのシーンの背景が実は絵だったの？ あのシーンが模型？ といった手品の種明かしの驚きの連続。帰って全作を見直したことはいうまでもない。(善)

▶今月は自転車が行方不明になった。コンタクトは下水へと流れていった。体調も崩した。誕生日がきた。プレゼントをもらった。アンミラでウエイトレスさんがケーキにろうそくを立ててくれた。ふっ、地獄と天国がこの世にあるものだったのね……。え？ これから本当地獄を見せてやるぜって？ え、遠慮しときます〜、あははのは。(て)

▶IBM用WORLD CIRCUITはAMIGA版より絵がきれいだし通信で対戦もできる。プリンスオブペルシャ2もIBMにだけ出てAMIGAには出る気配もない。最近ではゲームでもIBMが元氣。少し面白くない。ところで最近国産のレースシミュレーションを買ったら日本の技術レベルが見えて然とした。金返せ〜。

(教授の引退にショックを受けてしまったA.T.)

▶長く続いた連載もダラダラと続けるよりは、と終了してしまっ。「高校教師」は数回しか見られなかったのだけれども、「高校教師」CD-ROMは面白かった。単行本になった「パブリカ」は近年まれに見る比類なき面白さ、深さ、すごさの大傑作であった。僕もパソコンを道具だと思ったことは一度もない。オモチャであり道楽でありメディアだ。(K)

▶「コーラのMください」の言葉に店員は、不審な顔をした。ないものを注文したかと改めてメニューを見る。ちゃんと3種類あるじゃないか。あれ、SとLの中間はRだ。ということで「コーラのR」と訂正する。「レギュラーサイズですね」とその店員。普通はMでわかるよなあと思いつつ、さすが「Mに飽きたらL」のお店だと変な感心をした。(KO)

▶北海道へ出かけた。時間のないなか札幌の街をふらふらと歩いていた。のどが乾いてふと自動販売機を見てみると、メツツのガラナではないか。おお、懐かしい。そのあと、自販機を見つけたときに近寄っていると、リボンナポリンという飲料を発見。味のほどは、ファイブミの炭酸を強くしたような味。やっぱり地方に行ったら自販機かな。(高)

▶近所の公園の機関車のかたちをした遊具には、夜になると、紙袋を抱えて帰ってくる人がいる。どうやら住処にしているらしい。昼間はそこでは子供たちが遊んでいる。きちんと「住み分け」がなされているようだ。そういえば朝方あんなに飛び交っている鳥も、ほかの時間帯には見かけない。なわばりがあるのだろうか。生き物っておもしろい。(ふ)

▶今月は、スロットでハマリプレイ数を更新してしまっ。573プレイのハマリの末、ようやく集中を引いてくるが、30ゲームでバンク。そのあと、1341プレイハマってドッカ〜！ 結局4万円負け。ちくしょう、オリエントエクスプレスなんて、「SOREX」なんて大っ嫌いだ！ 僕ってやっぱり、ギャンブルには向いていないんだろうか。(J)

▶なぜスバルは遅いの？ 叫ばないケンなんて……。考えてみればストIIは凄かった。12億人以上を擁する中華人民共和国の、半数が女性でその半分が未婚とすると総勢3億人。それがいつしか「中国娘」という呼称はただひとりの意味するようになったのである。うーむ、凄。さらにダルシムである。インド国民約8億5千万人……(以下略)。(U)

▶この号が発売になるころは、ワールドカップアジア地区最終予選の真っただなかだ。なんとかが突破して、悲願の初出場を果たしてほしい。そして来年のOh!Xの誌面はサッカー一色。新作ゲーム「World Cup」が登場し、LIVE in '94ではサッカー関連の曲があふれ、表紙もサッカーネタにしてくださいなんて要望が殺到するに違いない。わくわく。(T)

## microOdyssey

「人に伝える」ということ。「伝える」を辞書でひいてみる。「必要な事柄を、人を介して知らせる」もうひとつの意は、「伝わるようにする」。後者について考えてみる。なにかひとつの事柄を誰かに伝えるとき、相手はそこにある真意を汲み取ってくれるだろうか？ もちろん、あうんの呼吸でわかってくれる人もいるかもしれない。ただ、そんな人ばかりではない。伝えたいことが伝わらずやきもきすることもしばしばある。

たとえば「～が好き」と人に伝えるとしよう。その言葉を聞いたとき、「好き」の程度を人はどこで判断するだろうか？ 「好き」なもの（人）についての賛辞などの冗舌な表現かもしれない。または、その人の口調や熱心さ、視線など言葉以外のものであるかもしれない。つまりその人の心のなかにあるもの（思い）がどう表現されているか、ということであろう。「ゲーム」もそんな表現手段のひとつではないだろうか。

私が初めてパソコンに触ったところ、中村光一氏という方が/O誌によく投稿をされていた。私が見たのは「ラリーX」というゲームであった。PC-8001用のプログラムだったが、当時マシン語の入力方法も知らぬ私と友人は、いろんなことを調べながらそのプログラムを打ち込んだ。入力が終わりプログラムが動いた瞬間は感動で熱くなった。グラフィックはもちろん音楽はゲームセンターのものより劣っていた（仕方ないけど）。それでもパソコンで再現された「ラリーX」から中村氏のゲームに対する熱い思いが十分に伝わってきた。その熱い思いが、いまの彼の地位（チュンソフト代表取締役：スーパーファミコンで「弟切草」「トルネコの大冒険」を発売しているところ。ちなみに後者は「ローグ」のようで最高！）を築いた要因のひとつではないかと思う。

そしてX68000のユーザーにはそんな熱い思いを持った人たちが多いような気がする（MZのユーザーももちろん）。本誌、10月号より新連載の「ハードコア3Dエクスタシー」、このなかでは横内氏と丹氏が3Dへの熱い思いを表現している。今月号の特集のなかでも、3Dへの熱い思いを語っている人がたくさんいる。表現する言葉は違えども、そんな熱い思いがきっと読者にも伝わると思う。

話は少し飛ぶが、いま私は、非常に不思議な感じがしている。初めてパソコンを購入したのはX1（マニアタイプ）、その次の購入はX68000 PROである。Oh!MZの頃からの読者ではあった。だがとてもまじめな読者とはいえない。アンケートもほとんど出したことなかったし。そして、ふと気がつくところになっていた。その私にはなにができるだろうか？ いまはまだ、がむしゃらにやっているだけなのだが、読者の皆さんになにを伝えていけるのだろうか？ ただひとつだけいえることは、昔、パソコンの「ラリーX」を初めて見た時、X1やX68000に触ったときに感じたあの「熱い思い」は忘れないということ。人になにかを伝えることはとても難しいけれど、自分がそれに熱くなることが人に伝える第一歩になるような気がするから。

今年の夏は寒かったけれど、心のなかだけは「終わりのない夏」（No End Summer）のように熱くありたい。（高）

# 1993年12月号11月18日(木)発売

## 特集 仮想遊技空間で遊ぶ

・悪魔城ドラキュラ完全攻略  
・特大ゲームレビュー

データショウ/エレクトロニクスショウレポート

製品レポート&活用プログラム

BJC-880Jカラーハードコピープログラム

全機種共通システム

エディタセンブラREDA再掲載

## バックナンバー常備店

東京	神保町	三省堂神田本店5F 03(3233)3312	船橋	リプロ船橋店 0474(25)0111
	//	書泉ブックマートB1 03(3294)0011	//	芳林堂書店津田沼店 0474(78)3737
	//	書泉グランデ5F 03(3295)0011	千葉	多田屋千葉セントラルプラザ店 043(224)1333
	秋葉原	T-ZONE 7Fブックゾーン 03(3257)2660	埼玉	川越 黒田書店 0492(25)3138
	八重洲	八重洲ブックセンター3F 03(3281)1811	川口	岩淵書店 0482(52)2190
	新宿	紀伊国屋書店本店 03(3354)0131	茨城	水戸 川又書店駅前店 0292(31)0102
	高田馬場	未来堂書店 03(3209)0656	大阪	北区 旭屋書店本店 06(313)1191
	渋谷	大盛堂書店 03(3463)0511	都島区	駿々堂京橋店 06(353)2413
	池袋	旭屋書店池袋店 03(3986)0311	京都	中京区 オーム社書店 075(221)0280
	八王子	くまざわ書店八王子本店 0426(25)1201	愛知	名古屋 三省堂名古屋店 052(562)0077
神奈川	厚木	有隣堂厚木店 0462(23)4111	//	パソコンZ上前津店 052(251)8334
	平塚	文教堂四の宮店 0463(54)2880	刈谷	三洋堂書店刈谷店 0566(24)1134
千葉	柏	新星堂カルチュエ5 0471(64)8551	長野	飯田 平安堂飯田店 0265(24)4545
			北海道	室蘭 室蘭工業大学生協 0143(44)6060

## 定期購読のお知らせ

Oh!Xの定期購読をご希望の方は綴じ込みの振替用紙の「申込書」欄にある「新規」「継続」のいずれかに○をつけ、必要事項を明記のうえ、郵便局で購読料をお振り込みください。その際渡される半券は領収書になっていますので、大切に保管してください。なお、すでに定期購読をご利用の方には期限終了の少し前にご通知いたします。継続希望の方は、上記と同じ要領でお申し込みください。

基本的に、定期購読に関することは販売局で一括して行っています。住所変更など問題が生じた場合は、Oh!X編集部ではなくソフトバンク販売局へお問い合わせください。

### 海外送付ご希望の方へ

本誌の海外発送代理店、日本IPS(株)にお申し込みください。なお、購読料金は郵送方法、地域によって異なりますので、下記宛必ずお問い合わせください。

日本IPS株式会社

〒101 東京都千代田区飯田橋3-11-6

☎03(3238)0700



11月号

■1993年11月1日発行 定価600円(本体583円)

■発行人 橋本五郎

■編集人 稲葉俊夫

■発売元 ソフトバンク株式会社

■出版事業部 〒103 東京都中央区日本橋浜町3-42-3

Oh!X編集部 ☎03(5642)8122

販売局 ☎03(5642)8100 FAX 03(5641)3424

広告局 ☎03(5642)8111

■印刷 凸版印刷株式会社

©1993 SOFTBANK CORP. 雑誌02179-11 本誌からの無断転載を禁じます。

落丁・乱丁の場合はお取り替えいたします。





# X68K-PPI

## 自作派御用達 8255コンパチボード

当社は博物館や科学館等の展示物(ハード・ソフト)を制作しています。

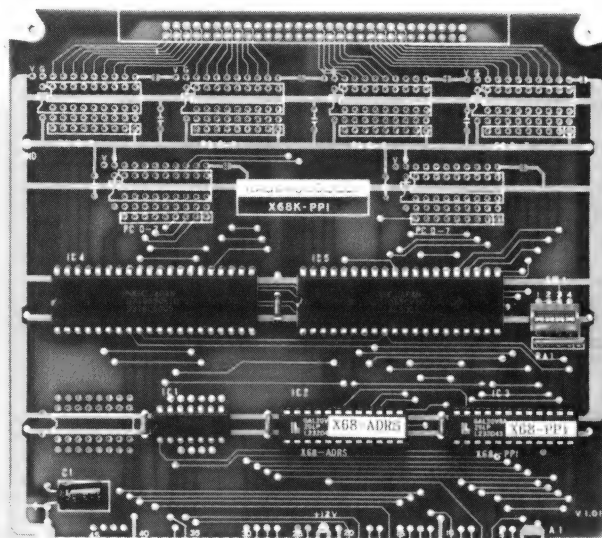
この技術と経験からX68シリーズ用I/Fボード

「X68K-PPI」を制作しました。

グラフィックや音楽と同期してソレノイドやモーターを動かすのに必要なインターフェースボードとして作られたのが「X68K-PPI」です。

- 48ビットI/Oボード。セミキット。
- μPD71055(8255コンパチ)2個搭載。
- 入出力用バッファICを搭載できるエリアを用意。(8ビット×6個分)
- X68030対応。
- 全回路図公開。使用しているGALの論理も公開。
- 定価22,000円(送料・税込み)

注意:本製品はセミキットです。入力出コネクターやバッファIC、プルアップ抵抗等は添付しておりません。ユーザーにて御用意をお願いします。  
(山-FAP-60-07.02B等。)半田付け作業が必要です。



# X68K-SCAN

## 電腦絵師に贈る スキャナボード

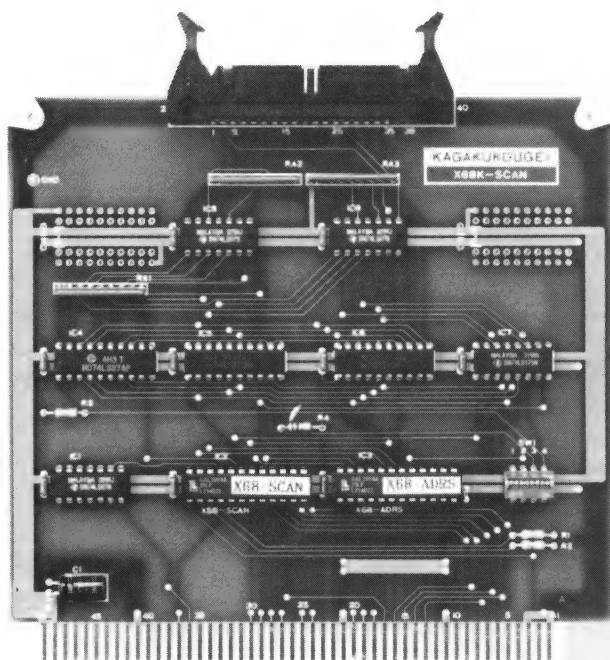
エプソンGTシリーズスキャナで高速入力を行うためのボードです。

X680x0の優れたグラフィックエディター「マチエール」

「Z's STAFF PRO-68K Ver. 3.0」で使えます。

(添付ソフト使用時。)

- エプソンGTシリーズスキャナ用パラレルボード。
- 接続ケーブル付き完成品。
- 「マチエール」「Z's STAFF PRO-68K Ver. 3.0」でパラレル入力ができるようにするソフト添付。(5/3.5インチ同梱)
- X68030対応
- 「マチエール」で512×512ドット6万5千色を1分強で入力。(X68030使用時。ちなみにRS-232C 19200bpsで7分17秒。当社測定)
- 対応スキャナ:エプソンGT-1000/4000/6000/6500/8000 (GT-6500にはエプソンのシリアル・パラレルボードGT65RSPRBが必要です。)
- 全回路図公開。ソフトはソースも添付。コピーフリー。
- 増設プリンターポート/汎用パラレル入出力ポートとしてもお使い頂けます。
- 定価29,000円(送料・税込み)



注意:シャープ製パラレルボードCZ-6BN1との互換性はありません。

「マチエール」は株サンワードの製品です。

「Z's STAFF PRO-68K」は株ツァイトの製品です。

### ＝通信販売の方法＝

ご注文は、住所・氏名(会社名)・TEL・品名・個数を明記の上、郵便振替か現金書留にてお願い致します。入金確認後発送いたします。現金書留の場合はおつりのないようお願いします。振替手数料・書留送料につきましてはお客様負担となります。(送料・消費税は代金に含む)その他技術的なご質問等FAX・郵便にて受付けております。

郵便振替:東京0-665905

株式会社 科学工学研究所

〒164 東京都中野区本町5丁目14番23号 TEL.03(5385)4651 FAX.03(5385)4650



# 私は、 680x0が 好きです。



## はじめまして！

当社は、X680x0シリーズ用のソフトウェアを開発・販売するためにできた、新しいソフトウェアハウスです。

当社のソフトウェア第1段はX680x0のソフトウェア開発を大幅に効率化するXCASAです。

既存の開発ツールに不満をもっている方は、ぜひカタログをご請求下さい。

価格 **¥19,800** (税込み)

本体 **¥19,224**

## Béシステム

〒151 東京都渋谷区本町2-33-20  
カーサヴェルデ102

TEL. **03-3372-5336**

FAX. **03-3372-5886**

**アルバイト募集中！**

●資格:18~28才の男女で週3回ぐらい出勤できる方 ●交通:京王線 初台駅より 徒歩5分  
●時間:AM10:00~PM 5:00 ●仕事内容:事務、電話の応対、製品の梱包 ●時給:¥800~1000

**P&Aならではの  
新品パソコン**

**5年  
保証**

お支払いは、  
便利な商品  
到着払い  
(手数料要)を  
ご利用  
下さい。

《業界No.1の"P&Aメンテナンスサポート"》

**最高の保証システム**

- ①業界最長の新品パソコン5年保証  
(※モニター・プリンター3年間保証!! ※一部商品は除きます。)
- ②中古パソコンの1年間保証  
(モニター・プリンター6ヶ月間保証)
- ③初期不良交換期間3ヶ月  
(※新品商品に限らせていただきます)
- ④永久買取保証
- ⑤配達指定OK!! (土曜・日曜・祭日もOK!!)
- ⑥夜間配送もOK!!  
(※PM6:00~PM8:00の間 ※一部地域は除きます。)

**便利でお得な支払いシステム**

- ①翌月一括払い手数料無料(ご利用下さい。)
- ②業界No.1の低金利
- ③月々の支払いは¥1,000より
- ④9ヶ月先からのスキップ払いOK!!
- ⑤84回までの分割、ボーナス併用OK!!
- ⑥カレッククレジット
- ⑦ステップアップクレジット
- ⑧ボーナスだけで10回払いOK!!
- ⑨現金一括払いOK!!
- ⑩商品到着払いOK!! (代引き手数料が必要になります。)

●法人向け  
リースシステム  
業務に最適なシ  
ステムを構築しま  
す。損金処理が可  
能なリース契約を  
どうぞ。

(※商品・金額ご確認の上、銀行振込・現金書留にてご入金下さい。)

SHARP=X68030エキスパートショップ

**P&A=X68030  
ダブルNEWフェア**  
《10月18日~11月17日》

**32ビット X68030いよいよ登場。  
購入ダブルチャンス!!**

**X68030発売記念**

X68030をモニターとセットで購入の方!!

さらに現在お持ちのパソコンと下取り交換されたお客様に期間中もれなく、

- ① サイバーステック..... (CZ-8NJ2 ¥23,800)
- ② X-68000フロッピーアタッシュケース(¥8,000)  
とクリスタルポルシェ(¥8,000)

以上のいずれかプレゼント!!



**今だからこそ選ぶズバリお買い得セット**

①



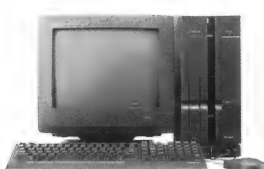
**通信セット**

- CZ-500C+CZ-608D
- MC-14400FX(FAXモデム、マイクロコア)
- CZ-257CSD(communication)

合計定価 ¥559,400

P&A超特価 **¥396,000**

②



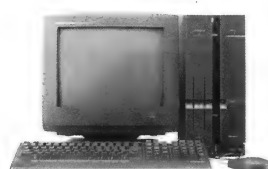
**ハードディスクセット**

- CZ-500C+CZ-608D
- LHD-FM100E(ロジテック、100MB)
- ケーブル付

合計定価 ¥598,600

P&A超特価 **¥396,800**

③



**MIDI ミュージックセット**

- CZ-500C+CZ-608D
- SX-68MII(システムサコム)
- CM-300(ローランド)

合計定価 ¥570,600

P&A超特価 **¥406,000**

※本広告の掲載の商品の価格については、消費税は含まれておりません。

注目!! 冬のボーナス一括払い手数料(金利)無料(平成5年11月末/12月末/平成6年1月末のいずれかを指定ください。)



# P&A

# 全国通販

★頭金なし!!  
★即日発送!!

## 32ビットX68030いよいよ登場(送料¥2,000・消費税別)

④

### グラフィックセット

- CZ-500C+CZ-608D
- HS-7RII(オムロン、スキャナ)
- Z's STAFF PRO-68K Ver.3.0 (ツァイト)



合計定価 ¥590,600

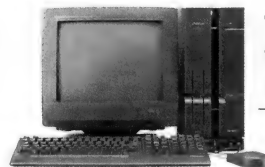
P&A超特価

**¥411,000**

⑤

### 光磁気ディスクセット

- CZ-500C+CZ-608D
- CS-M120(コパル、光磁気)
- ターミネータ、ケーブル付
- MOカートリッジ(1枚)



合計定価 ¥670,800

P&A超特価

**¥477,000**

### 本体の変更

- ① CZ-510Cに変更の場合 ¥71,000
  - ② CZ-300Cに変更の場合 ¥1,000
  - ③ CZ-310Cに変更の場合 ¥64,000
- 加算して下さい。

### モニターの変更

- ① CZ-607D(チューナー付)に変更の場合 ¥3,000
  - ② CZ-614D(チューナー付)に変更の場合 ¥31,000
  - ③ CU-21MD 加算して ¥60,000 下さい。
- ※300シリーズにチューナー付のモニターを接続の場合CRTケーブルを購入して下さい。

## X68000 Compact XVI

旧シリーズ今が買いどき!!  
(クレジット表:送料・消費税込み)  
送料 ¥2,000・消費税別

### ① 本体+モニター

- CZ-674C-H
  - CZ-608D-H
- 定価 ¥392,800



P&A超特価 **¥162,000**

12回 14,800 24回 7,800 36回 5,400 48回 4,300 60回 3,600

### ② 本体+モニター+FDD(5"×2)

- CZ-674C-H
  - CZ-608D-H
  - CZ-6FD5(FDD)
- 定価 ¥492,600



P&A超特価 **¥209,000**

12回 19,100 24回 10,100 36回 7,000 48回 5,500 60回 4,600

### ③ 本体+モニター(TVチューナー付)

- CZ-674C-H
  - CZ-614D-TN
  - CZ-6CR1(RGBケーブル)
  - CZ-6CT1(TVコントロール)
- 定価 ¥443,000



P&A超特価 **¥199,000**

12回 18,200 24回 9,600 36回 6,700 48回 5,200 60回 4,400

### ④ 本体+モニター(TVチューナー付)+FDD(5"×2)

- CZ-674C-H
  - CZ-614D-TN
  - CZ-6CR1(RGBケーブル)
  - CZ-6CT1(TVコントロール)
  - CZ-6FD5(FDD)
- 定価 ¥542,800



P&A超特価 **¥247,000**

12回 22,500 24回 11,900 36回 8,300 48回 6,500 60回 5,400

## X68000 XVI

### ① 本体+モニター

- CZ-634C-TN(本体)
  - CZ-608D-H(モニター)
- 定価 ¥462,800



P&A超特価 **¥213,000**

12回 19,500 24回 10,300 36回 7,100 48回 5,600 60回 4,700

### ② 本体+モニター

- CZ-634C-TN(本体)
  - CZ-614D-TN(モニター)
- 定価 ¥503,000



P&A超特価 **¥243,000**

12回 22,200 24回 11,700 36回 8,100 48回 6,400 60回 5,300

※ Compact XVI/XVI ①、②のモニターを CZ-607D-TN(定価 ¥99,800)に変更の場合 ¥3,000 加算して CZ-621D(B)(定価 ¥168,000)に変更の場合 ¥58,000 下さい。

# P&A

株式会社ピー・アンド・エー  
〒124 東京都葛飾区新小岩2丁目2番地20号

☎ 03-3651-0148(代) 03-3651-0141

●営業時間:  
AM10:00~PM7:00  
日・祭:  
AM10:00~PM6:00  
●定休日/毎週水曜日  
●FAX:  
03-3651-0141

●価格は流通事情により変動致すので、銀行振込・書留等の送付前にあらかじめお電話にてご確認ください。

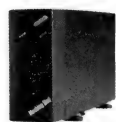
## カラーイメージスキャナー



◎ JX-220X **限定10台**  
(RS-232Cケーブル・ユーティリティソフト付)  
定価 ¥168,000 ▶ 特価 **¥99,800**

## X68000/68030 専用ハードディスク

外



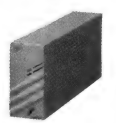
■富士通  
◎ FMHD-1201G(120MB、17ms)  
.....定価 ¥70,000 ▶ 特価 **¥49,800**  
◎ HD-K200A(モックンボード)(200MB、13ms)  
.....定価 ¥79,800 ▶ 特価 **¥57,000**

付



■ロジテック  
◎ SHD-FMX120(120MB)(ケーブル付)  
.....定価 ¥59,800 ▶ 特価 **¥47,000**  
◎ SHD-FMX240(240MB)(ケーブル付)  
.....定価 ¥138,000 ▶ 特価 **¥62,000**

内



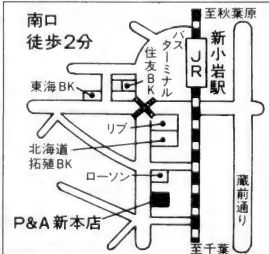
■ジェフ  
◎ GF-200(200MB、15ms、64K).....定価 ¥98,000 ▶ 特価 **¥59,000**  
◎ GF-240e(240MB、15ms、64K).....定価 ¥118,000 ▶ 特価 **¥63,500**  
◎ GF-340i(340MB、14ms、64K).....定価 ¥158,000 ▶ 特価 **¥87,800**  
◎ GF-540i(540MB、8.5ms、256K).....定価 ¥238,000 ▶ 特価 **¥151,800**

(銀行振込でお申し込みの方)(電信扱いでお振込み下さい。)

〔振込先〕さくら銀行 新小岩支店  
当座預金 2408626  
(株)ピー・アンド・エー

### 超低金利クレジット率

回数	3	6	10	12	15
手数料	2.9	3.9	4.9	5.4	8.4
回数	24	36	48	60	72
手数料	11.4	15.9	20.9	26.9	34.9



※お支払いは、便利な商品到着払い(手数料要)をご利用下さい。

※お支払いは、便利な商品到着払い(手数料要)をご利用下さい。

# P&Aならではの 新品パソコン 5年保証

## 《業界No.1の"P&Aメンテナンスサポート” 最高の保証システム

- ①業界最長の新品パソコン5年保証  
(※モニター・プリンター3年間保証!! ※一部商品は除きます。)
- ②中古パソコンの1年間保証  
(モニター・プリンター6ヶ月間保証)
- ③初期不良交換期間3ヶ月  
(※新品商品に限らせていただきます)
- ④永久買取保証
- ⑤配達指定OK!!(土曜・日曜・祭日もOK!!)
- ⑥夜間配送もOK!!  
(※PM6:00~PM8:00の間 ※一部地域は除きます。)

## 便利でお得な支払いシステム

- ①翌月一括払い手数料無料(ご利用下さい。)
- ②業界No.1の低金利
- ③月々の支払いは¥1,000より
- ④9ヶ月先からのスキップ払いOK!!
- ⑤84回までの分割、ボーナス併用OK!!
- ⑥カレッククレジット
- ⑦ステップアップクレジット
- ⑧ボーナスだけで10回払いOK!!
- ⑨現金一括払いOK!!
- ⑩商品到着払いOK!!(代引き手数料が必要になります。)

(※商品・金額  
ご確認の上、  
銀行振込・現  
金書留にてご  
入金下さい。)

**モデム** (送料 ¥1,000)

マイクロコム MC-14400FX  
(定価 ¥46,800)  
特価 ¥34,500

富士通 FMMD-3111G  
(定価 ¥35,800)  
特価 ¥24,800

オムロン MD-24XT10V  
(定価 ¥29,800)  
特価 ¥22,800

MD-96XT10V  
(定価 ¥46,800)  
特価 ¥35,500

アイワ PV-AF144V5  
(定価 ¥64,800)  
特価 ¥49,000

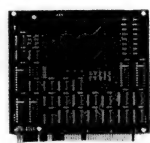
- お近くの方は、お立寄下さい。専門係員が説明いたします。
- 本体単品でも受付します。詳しくは、お電話にてお問合せ下さい。

## 周辺機器コーナー (送料 ¥1,000・消費税別)

1 BF-68PRO.....定価 ¥19,800▶特価 ¥14,400	15 CZ-6BG1.....定価 ¥59,800▶特価 ¥43,000
2 CZ-8NM3.....定価 ¥9,800▶特価 ¥7,200	16 CZ-6BU1.....定価 ¥39,800▶特価 ¥28,500
3 CZ-8NT1.....定価 ¥13,800▶特価 ¥10,000	17 CZ-6PV1.....定価 ¥198,000▶特価 ¥142,000
4 CZ-6BE2A.....定価 ¥59,800▶特価 ¥42,800	18 CZ-6BS1.....定価 ¥29,800▶特価 ¥21,500
5 CZ-6BE2B.....定価 ¥54,800▶特価 ¥39,300	19 CZ-8NJ2.....定価 ¥23,800▶特価 ¥17,500
6 CZ-6BE2D.....定価 ¥54,800▶特価 ¥39,300	20 CZ-6BL2.....定価 ¥298,000▶特価 ¥214,000
7 CZ-6BF1.....定価 ¥49,800▶特価 ¥35,800	21 CZ-6CSI(674C用).....定価 ¥12,000▶特価 ¥8,900
8 CZ-6BP1.....定価 ¥79,800▶特価 ¥57,000	22 CZ-68HA.....▶特価 ¥91,000
9 CZ-6BM1.....定価 ¥26,800▶特価 ¥19,300	23 CZ-6CR1(RGBケーブル).....定価 ¥4,500▶特価 ¥3,600
10 AN-S100.....定価 ¥36,600▶特価 ¥26,300	24 CZ6CT1(テレビコンロー).....定価 ¥5,500▶特価 ¥4,400
11 CZ-6SD1.....定価 ¥44,800▶特価 ¥32,500	25 CZ-6BP2.....定価 ¥45,800▶特価 ¥33,300
12 CZ-6BN1.....定価 ¥29,800▶特価 ¥21,500	
13 CZ-6BV1.....定価 ¥21,000▶特価 ¥15,200	
14 CZ-6BC1.....定価 ¥79,800▶特価 ¥57,000	

- システムサコムボード
- SX-68MII (MIDI)  
定価 ¥19,800▶特価 ¥13,500
  - SX-68SC (SCSI)  
定価 ¥26,800▶特価 ¥17,500

## X 68030/68000 メモリボード (I/Oデータ)



- ①SH-5BE4-8M(X68030用)  
(送料・消費税込み ¥47,586) 特価 ¥45,500
- ②SH-6BE1-1ME(600C専用)  
(送料・消費税込み ¥12,669) 特価 ¥11,600
- ③1MB増設RAMボード(ACE/PRO/PROII用)  
(送料・消費税込み ¥12,669) 特価 ¥11,600
- ④2MB増設RAMボード(拡張スロット用)  
(送料・消費税込み ¥24,411) 特価 ¥23,000
- ⑤4MB増設RAMボード(拡張スロット用)  
(送料・消費税込み ¥40,170) 特価 ¥38,300

## X68000用ソフトコーナー

◆Z'sSTAFFPRO68KVer.3.0(ツアイト).....	定価 ¥58,000▶特価 ¥37,500
◆Z'sTRIPHONYデジタルクラフト(ツアイト).....	定価 ¥39,800▶特価 ¥27,000
◆テラツォ(ハミングバード).....	定価 ¥19,400▶特価 ¥13,600
◆ラジックパレット(ミュージカルプラン).....	定価 ¥19,800▶特価 ¥14,200
◆たーみのる2(SPS).....	定価 ¥17,800▶特価 ¥13,000
◆Mu-1Super(サンワード).....	定価 ¥39,800▶特価 ¥28,500
◆CMA68K(シティソフト).....	定価 ¥29,000▶特価 ¥21,800
◆サイクロンEXPRESSα68.....	定価 ¥98,000▶特価 ¥69,000
◆C-TRACE68Ver.3.0(キャスト).....	定価 ¥98,000▶特価 ¥68,500
◆OS-9/X68030 V.2.4.5(マイクロウェアシステムズ).....	定価 ¥25,000▶特価 ¥19,900
◆C&ProfessionalPackV3.2(マイクロウェアジャパン).....	定価 ¥80,000▶特価 ¥57,800
◆ウエイトペイント1~3(ウエイトレイン)[各].....	定価 ¥15,000▶特価 ¥11,500
◆マチエール Ver.2.0.....	定価 ¥39,800▶特価 ¥28,800
◆WindexPRO68(JEL).....	定価 ¥28,000▶特価 ¥20,500
◆CZ-213MSDMUSICPRO68K.....	定価 ¥18,800▶特価 ¥13,200
◆CZ-214MSDSOUNDPRO68K.....	定価 ¥15,800▶特価 ¥11,300
◆CZ-215MSDSamplingPRO68K.....	定価 ¥17,800▶特価 ¥12,500
◆CZ-220BSDDATAAPRO68K.....	定価 ¥58,000▶特価 ¥40,000
◆CZ-225BSV Multiword Ver.2.0.....	定価 ¥32,000▶特価 ¥23,000
◆CZ-243BSDCYBERNOTEPRO68K.....	定価 ¥19,800▶特価 ¥15,000

## 周辺機器特選品コーナー

<b>JX-325X</b> カラーイメージスキャナ 定価 ¥190,000 特価 ¥143,000	<b>CZ-6VTI</b> カラーイメージユニット 定価 ¥69,800 特価 ¥49,500	<b>CZ-6TU</b> RGBシステムチューナー 定価 ¥33,100 特価 ¥23,900
<b>JX-32F12</b> (写真上部分) 定価 ¥148,000 特価 ¥112,000		
<b>JX-220X</b> カラーイメージスキャナ 《限定》 定価 ¥168,000 特価 ¥99,800	<b>CZ-8NSI</b> カラーイメージスキャナ 定価 ¥188,000 特価 ¥133,000	(X68030用) 増設RAMボード & 数値演算プロセッサ <b>CZ-5BE4</b> 定価 ¥54,800 特価 ¥42,000 <b>CZ-5ME4</b> 定価 ¥49,800 特価 ¥38,000 <b>CZ-5MPI</b> 定価 ¥54,800 特価 ¥42,000

FDD(5インチ×2基)

■CZ-6FD5  
(シャープ)  
(定価 ¥99,800)  
P&A超特価  
¥49,800

**プリンター**  
(ケーブル用紙付・送料 ¥1,000・消費税別)

■CZ-8PC5-BK  
定価 ¥96,800  
特価 ¥53,000

■CZ-8PK10  
定価 ¥97,800  
特価 ¥71,000

**カラーイメージジェット**

■IO-735X-B  
定価 ¥248,000  
特価 ¥135,000  
(送料・消費税込み ¥140,080)

## P&A特選パソコンラック&OAチェア (消費税込み)(送料無料、離島を除く)

① 3段 ¥8,240	② 4段 ¥9,785	③ 5段 ¥11,845	④ ¥9,270 (ダークグレー) ●ガスシンダー
(W-640)	(W-640)	(W-640)	(W-640)
※全機種→キャスター付 ※上から2番目棚板移動可能(4/5段) 4段→黒、3/5段→ホワイト			⑤ ¥13,390 (ダークグレー) ●ガスシンダー ●肘付

(送料 ¥700・消費税別)

注目!! 冬のボーナス一括払い手数料(金利)無料 (平成5年11月末/平成6年1月末のいずれかを指定ください。)

◆CZ-247MSDMUSICPRO68K(MID).....	定価 ¥28,800▶特価 ¥20,500
◆CZ-249GSDCANVASPRO68K.....	定価 ¥29,800▶特価 ¥22,000
◆CZ-251BSDHyperword.....	定価 ¥39,800▶特価 ¥29,400
◆CZ-253BSDCARDPRO68KVer.2.0.....	定価 ¥29,800▶特価 ¥22,700
◆CZ-257CSDCommunicationPRO68KVer.2.0.....	定価 ¥19,800▶特価 ¥15,300
◆CZ-258BSDTeleportationPRO68K.....	定価 ¥22,800▶特価 ¥16,900
◆CZ-261MSDMUSICstudioPRO68KVer.2.0.....	定価 ¥28,800▶特価 ¥21,200
◆CZ-263GWD EasypaintSX-68K.....	定価 ¥12,800▶特価 ¥9,800
◆CZ-264GWD Easydraw SX-68K.....	定価 ¥19,800▶特価 ¥15,300
◆CZ-265HSDNewPrintShopVer.2.0.....	定価 ¥20,000▶特価 ¥15,400
◆CZ-266BSDPressConductorPRO68K.....	定価 ¥28,800▶特価 ¥22,000
◆CZ-267BSDCHARTPRO68K.....	定価 ¥38,000▶特価 ¥29,800
◆CZ-272CWCCommunicationSX68K.....	定価 ¥19,800▶特価 ¥14,500
◆CZ-275MWDSDSOUNDSX68K.....	定価 ¥15,800▶特価 ¥11,500
◆CZ-284SSDOS-9/X68000Ver.2.4.....	定価 ¥35,800▶特価 ¥25,600
◆CZ-286BSDBUSINESSPRO68KPopular.....	定価 ¥28,000▶特価 ¥20,500
◆CZ-288LWD開発キット(workroom).....	定価 ¥39,800▶特価 ¥29,700
◆CZ-290TWD SX-WINDOW ディスクアクセサリ集.....	定価 ¥14,800▶特価 ¥11,500
◆CZ-294SS(5'')/SSC(3.5'') SX-WINDOW Ver.3.0.....	定価 ¥19,800▶特価 ¥15,200
◆CZ-295LSD C-Compiler PRO68K Ver.2.1 NEW KIT.....	定価 ¥44,800▶特価 ¥32,500

☆ゲームソフト25%OFF OK!!(一部ソフト除く)



注目!!

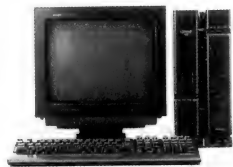
★中古パソコン1年間保証システム!!  
(※モニター、プリンター6ヶ月間保証)

高価

高価

中古その場で現金買取り下取りOK!! 電話一本ですぐ買える!  
中古パソコンはP&Aにおまかせ!!

## P&A特選今月中古特選品



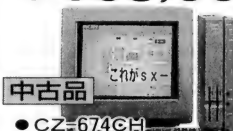
- CZ-600C.....¥55,000
  - CZ-601C.....¥65,000
  - CZ-611C.....¥70,000
  - CZ-652C.....¥75,000
  - CZ-612C.....¥95,000
  - CZ-603C.....¥85,000
  - CZ-653C.....¥78,000
  - CZ-612C.....¥ 90,000
  - CZ-623C.....¥110,000
  - CZ-674C.....¥108,000
  - CZ-634C.....¥130,000
  - CZ-644C.....¥178,000
- (上記は単品価格、モニター別売)

新古品

限定

- CZ-674CH
- CZ-608DH

¥168,000



- CZ-674CH
- 68000専用モニター付

¥138,000

新古品

限定

- CZ-634CTN(チタン)(中古)
- CZ-613D(グレー)(新品)

¥200,000



- CZ-634CTN
- 68000専用モニター付

¥163,000

新古品

限定

- CZ-644CTN
- CZ-604DB

¥248,000



- CZ-644CTN
- 68000専用モニター付

¥213,000

### グレードアップ

現在お持ちのパソコンとX68030シリーズを下取り交換されたお客様に期間中もれなく!

- ①サイバーステック (CZ-8NJ2 ¥23,800)
- ②X-68000フロッピーアタッシュケース (¥8,000) とクリスタルポルシェ (¥8,000)

以上のいずれかプレゼント!!

①



②



## グレードアップ差額表

新品	CZ-500CB	(80MB HD内蔵) CZ-510CB	CZ-300CB	CZ-310CB
下取				
CZ-674C	¥195,000	¥263,000	¥190,000	¥250,000
634C	¥175,000	¥243,000	¥170,000	¥230,000
644C	¥125,000	¥193,000	¥120,000	¥180,000
623C	¥205,000	¥273,000	¥200,000	¥260,000
653C	¥255,000	¥323,000	¥240,000	¥300,000
604C	¥225,000	¥293,000	¥230,000	¥290,000
603C	¥255,000	¥323,000	¥250,000	¥310,000
602C	¥255,000	¥323,000	¥250,000	¥310,000
601C	¥265,000	¥333,000	¥250,000	¥310,000
600C	¥275,000	¥343,000	¥260,000	¥320,000
611C	¥255,000	¥323,000	¥240,000	¥300,000
612C	¥245,000	¥313,000	¥240,000	¥300,000
613C	¥235,000	¥303,000	¥240,000	¥300,000
PC-9801RX2	¥245,000	¥313,000	¥240,000	¥300,000
DA2	¥215,000	¥283,000	¥210,000	¥270,000

※お支払いは、便利な商品到着払い(手数料要)をご利用下さい。

### 中古・高価現金買取り/下取りOK!!

■まずはお電話下さい。  
下取り専用買取り電話 ▶ **03-3651-1884** FAX. 03-3651-0141

■下取り・買取りで、お急ぎの方は、直接当社に来店、または宅急便にてお送りください。

買取り価格...完動品・箱/マニュアル/付属品の価格です。

- 下取りの場合...価格は常に変動していますので査定額を電話で確認してください。(差額は、P&A超低金利クレジットをご利用ください。)
- 買取りの場合...現品が着き次第、2日以内に高価買取金額を連絡し、振込み、又は書留でお送り致します。
- 近郊の方はP&A本店に直接お持ちください。即金にて¥1,000,000までお支払い致します。

- 最新の在庫情報・価格はお電話にてお問い合わせください。
- 買い取りのみ、または、中古品としての交換も致します。詳しくは電話にて、お問い合わせください。
- 価格は変動する場合がございますので、ご注文の際には必ず在庫をご確認ください。
- 本商品の掲載の商品の価格については、消費税は、含まれておりません。
- 現金書留及び銀行振込でお申し込みの方は、上記商品の料金に3%加算の上でお申し込み下さい。詳しくは、お電話でお問い合わせください。

### 《便利な超低金利クレジットをご利用ください》

- 月々¥1,000円からOK!!
- ボーナス払いOK!! (夏冬10回までOK)
- 支払い回数1回~84回
- お払いは、8ヶ月先からでもOK!!

### 通信販売お申し込みのご案内

[現金一括でお申し込みの方]

- 商品名およびお客様の住所・氏名・電話番号をご記入の上、代金を当社まで、現金書留でお送りください。(プリンター・プロッシーの場合、本体使用機種名をご明記のこと)

[銀行振込でお申し込みの方]

- 銀行振込ご希望の方は必ずお振込みの前にお電話にてお客様の住所・お名前・商品名等をお知らせください。

(電信扱いでお振込みください。)

[クレジットでお申し込みの方]

- 電話にてお申し込みください。クレジット申し込み用紙をお送りいたしますので、ご記入の上、当社までお送りください。

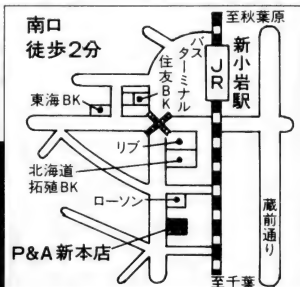
- 現金特別価格でクレジットが利用できます。残金のみに金利がかかります。

- 1回~84回払いまで出来ます。但し、1回のお支払い額は¥1000円以上。

〔振込先〕さくら銀行 新小岩支店  
当座預金 2408626 ㈱ピー・アンド・エー

### 超低金利クレジット率

回数	3	6	10	12	15	24	36	48	60	72
手数料	2.9	3.9	4.9	5.4	8.4	11.4	15.9	20.9	26.9	34.9



マイコン  
専門  
ショップ

P&A

株式会社ピー・アンド・エー  
〒124 東京都葛飾区新小岩2丁目2番地20号

☎ 03-3651-0148(代) FAX. 03-3651-0141

営業時間  
平日: AM10:00~PM7:00  
日祭: AM10:00~PM6:00

●現金書留及び銀行振込でお申し込みの方は、上記商品の料金に3%加算の上でお申し込みください。詳しくは、お電話でお問い合わせください。

恒例

# SHARP わんさかフェア in TSUKUMO

ツクモグローバルカード

好評  
入会者受付中/  
18才以上なら  
学生でもOK!!  
～国内・海外でも使える多機能カード～  
ジャックス・VISA・セントラル・マスターのカード  
です。分割払い、ボーナス払いもOK! クレジット  
申し込みと同時にカード申し込みOK。  
お申し込みは☎03(3251)9898又は各店で  
★各店舗では、JCB、日本信託、DC他各種カードも取り扱っております

11月27日(土)・28日(日)

冬のボーナス先取りで、  
安いのに親切なTSUKUMOに行こう!!

X680x0シリーズ周辺機器・特価品

ツクモパソコン本店Ⅱ3F  
ツクモニューセンター店

液晶ビューカム・プロジェクタ等映像関係

ツクモ5号店 (AVを中心に、業務用プロビデオ・CS・BSなどを取り扱っております。)

ワープロ・FAX・電子手帳・液晶ペンコムなど

ツクモパソコン本店ⅡB1F

大好評  
発売中!!

ツクモ特価  
販売中!

シリーズ最高峰。  
ユーザーの  
期待に応えて  
更にパワーアップした  
X68030/

X68030

- 新たに32ビットCPU(MC68EC030/25MHz)を搭載し、従来機の2.4～4.2倍以上のスピードアップを実現!
- 成熟するウィンドウ環境、使いやすさと高性能を追求し、動画機能、SX-WINDOW Ver3.0搭載
- SX-WINDOWの操作環境を考え、4MBメモリ内蔵
- カラー液晶ディスプレイ接続可能

5インチFDDモデル	CZ-500C-B	定価 ¥398,000
5インチHDDモデル	CZ-510C-B	定価 ¥488,000
3.5インチFDDモデル	CZ-300C-B	定価 ¥388,000
3.5インチHDDモデル	CZ-310C-B	定価 ¥478,000

超速!

X68030用8MB  
増設RAMボード発売!

●これ1枚でいっきに12MBフル実装

SH-5BE4-8M  
ツクモ  
特価 ¥46,800

おすすめの  
組み合わせ!!

→ハードディスクと...  
CZ-500C-B ..... ¥398,000  
240MBハードディスク ..... サービス

ツクモ特価 ¥360,000

- クレジット例(36回払・税込み)  
初回 ¥15,201 + 月々 ¥12,300 × 35回

→X68000の5インチモデルをお持ちの方には...  
CZ-300C-B ..... ¥388,000  
TS-XFDCA ..... ¥9,800

合計定価 ¥397,800  
ツクモ特価 ¥298,000

- クレジット例(24回払・税込み)  
初回 ¥16,415 + 月々 ¥14,700 × 23回

満開製作所の商品も取り扱っております。

●X68000 CompactXVI 24MHz改  
RED ZONE ..... ツクモ特価 ¥160,000  
RED ZONE+MK-FD1 ツクモ特価 ¥180,000

●満開製5インチFDD  
MK-FD1 ..... ツクモ特価 ¥39,800  
MK-FD1カラーリングモデル ツクモ特価 ¥44,800

通信販売のご注文は下記フリーダイヤルへ。

全国どこからでも通話料無料

愛・注・専・用  
フリーダイヤル 0120-377-999

通販センター 03-3251-9911

商品についてのお問い合わせは各店又は通販へ。

クレジット払い

月々 ¥3,000以上の均等払いも頭金なし、夏・冬ボーナス2回払いも受付中!

カード払い(¥5,000以上)

通信販売での利用カード、ツクモグローバルカード、VISAカード、セントラル・ジャックス・マスターカードより電話で通販部へお申し込み下さい

各種リース払い

くわしくは各店にお問い合わせ下さい。ケースに合わせてご相談にのります。

全国代金引き換え配達

お申し込みは☎03-3251-9911へお電話1本! 配達日の指定もできます。

現金書留払い

〒101-91 東京都千代田区神田郵便局私書箱135号ツクモ通販センター Oh/X係

銀行振込払い

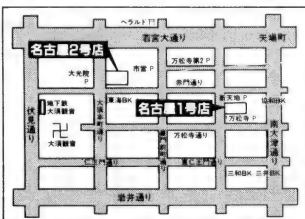
事前に書でお届け先をご連絡下さい。三和銀行 秋葉原支店(書)1009939 ツクモテンキ

ツクモin名古屋

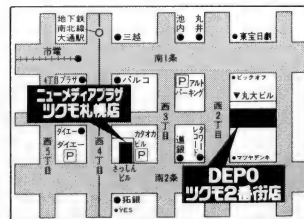
〈1号店 第一アメ横ビル内〉  
〈2号店 第二アメ横ビル内〉

ツクモin札幌

ツクモ札幌店  
〈DEPOツクモ2番街店〉



名古屋1号店 ☎052(263)1655 休毎週火曜日  
名古屋2号店 ☎052(251)3399 休毎週水曜日



札幌店 ☎011(241)2299 休毎週木曜日  
DEPO店 ☎011(242)3199 休毎週木曜日



# ツクモオリジナルTSシリーズ

目のつけどころがツクモでしょ。

TS-3XR  
シリーズ!!

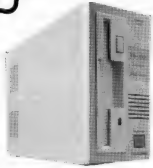
●X68000 & X68030シリーズ対応3.5インチフロッピーディスクドライブ

〈仕様〉  
●3.5インチ2DD/2HD/2HCフォーマット対応  
●ユーティリティソフト付属  
(デバイスドライバー/フォーマッター)  
●標準サイズケーブル付

TS-3XR1(1ドライブ) ツクモ特価  
定価 ¥44,800..... ¥34,800

TS-3XR2(2ドライブ) ツクモ特価  
定価 ¥57,800..... ¥45,800

●Compact XVI/X68030シリーズでお使いの方は、別売ケーブル(TS-XR5CA特価 ¥6,800)が必要です。



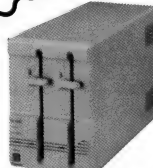
TS-5XR  
シリーズ!!

●X68000 Compact & 68030シリーズ対応フロッピーディスクドライブ

〈仕様〉  
●5インチ2HD/2DDフォーマット対応  
●ドライブ番号切り換えスイッチ付  
●Compact XVI/X68030用ケーブル付

TS-5XR1(1ドライブ) ツクモ特価  
定価 ¥53,800..... ¥35,800

TS-5XR2(2ドライブ) ツクモ特価  
定価 ¥72,800..... ¥47,800



## MIDIコンピュータミュージック特選セット

### RolandセットA

◆SC-55MK II ¥69,000  
◆SX-68M II ¥19,800  
◆Mu-IGS ¥28,000 ツクモ特価

合計定価 ¥116,800 **¥92,000**

### RolandセットB

◆CM-500 ¥115,000  
◆SX-68M II ¥19,800  
◆Mu-IGS ¥28,000 ツクモ特価

合計定価 ¥162,800 **¥135,000**

### KORGセットA

◆AG-10 ¥49,000  
◆SX-68M II ¥19,800  
◆Mu-IGS ¥28,000 ツクモ特価

合計定価 ¥96,800 **¥82,000**

### KORGセットB

◆05R/W ¥69,000  
◆SX-68M II ¥19,800  
◆Mu-IGS ¥28,000 ツクモ特価

合計定価 ¥116,800 **¥92,000**

## 大容量記憶装置

### MO特選セット

Logitec	富士通OA	SONY
LMO-FMX330TS..... ¥178,000	CS-M120WA(Ⅱ)..... ¥178,000	RMO-S360..... ¥169,000
MOメディア..... サービス	SCSIケーブル..... サービス	MOメディア..... S360に同梱
SCSIケーブル..... サービス	ターミネータ..... サービス	SCSI ケーブル..... サービス
ツクモ特価	ツクモ特価	ツクモ特価
<b>¥128,000</b>	<b>¥128,000</b>	<b>¥138,000</b>

◆ハードディスク ※540MBはSCSIインターフェース内蔵機種のみ対応

120MBハードディスク(VIP-120CX).....	ツクモ特価 ¥39,800
240MBハードディスク(VIP-240CX).....	ツクモ特価 ¥58,000
340MBハードディスク(VIP-350CX).....	ツクモ特価 ¥85,000
540MBハードディスク.....	ツクモ特価 ¥128,000

## その他周辺機器

### ◆X680x0シリーズ用CD-ROMセット

CD-ROM ドライブ本体..... 定価 ¥89,800  
CD-ROM Driverソフト..... 定価 ¥4,800

合計定価 ¥94,600

※ケーブル・ターミネータはオプションです。別途購入願います。

ツクモ特価 **¥72,000**

### ◆スキャンコンバータユニット

電波XVGA-1V..... ツクモ特価 **¥59,300**

23%OFF

## コンピュータアート

### ◆スーパーグラフィックツールセット

その1. 慣れてしまうとマウスがいらない  
NS CalcComp Drawing Pad (タブレットセット)..... ¥76,500  
サンワード Matier (マチエール)..... ¥39,800

ツクモ特価 **¥95,000** 合計定価 ¥116,300

その2. ハイクオリティなのにこんなに安い

ビューレットバックカードHP Desk Jet 505J(インクジェット) ¥99,800  
ビューレットバックカードカラーキット..... ¥12,000  
アーベルプリンターケーブル..... ¥4,800  
サンワード Matier (マチエール)..... ¥39,800

ツクモ特価 **¥95,000** 合計定価 ¥156,400

## コンピュータアート

- 48ドットカラー熱転写プリンター
- CZ-8PC5-BK(限定品)..... ツクモ特価 ¥39,800
- カラーイメージジェット
- IO-735X-B..... ツクモ特価 ¥130,000
- バブルジェットプリンター
- BU-10V Lite(ケーブルセット)..... ツクモ特価 ¥41,800
- キャンノバブルジェット(A3縦対応)
- BU-220JC(ケーブル付)..... ツクモ特価 ¥72,800
- カラーイメージスキャナー
- CZ-8NS1..... ツクモ特価 ¥99,800
- JX-220X..... ツクモ特価 ¥135,000
- JX-325X..... ツクモ特価 ¥152,000
- カラーイメージユニット
- CZ-6VT1(BK)..... ツクモ特価 ¥49,000
- RGBシステムチューナー
- CZ-6TU(BK)..... ツクモ特価 ¥23,900

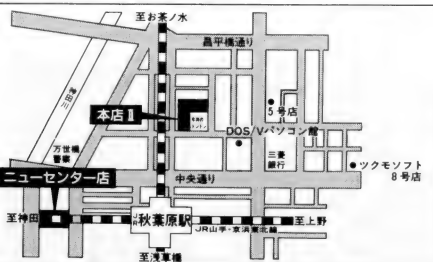
## X680x0シリーズ用RAMボード

●CZ-600C専用	ツクモ特価	●X68030シリーズ用	ツクモ特価
SH-8BE1-1ME	¥11,000	SH-5BE4-8M	¥46,800
●ACE/PRO/PRO2シリーズ用	ツクモ特価	●XVI専用	ツクモ特価
PIO-8BE1-AE	¥11,000	CZ-8BE2A	¥42,500
●拡張スロット用	ツクモ特価	●Compact XVI専用	ツクモ特価
PIO-8BE2-2ME	¥23,000	CZ-8BE2D	¥39,000
●拡張スロット用	ツクモ特価	●CZ-8BE2A/D用拡張RAM	ツクモ特価
PIO-8BE4-4ME	¥39,000	TS-8BE2B	¥29,800

## パソコン通信

- ★ファックスモデム
- AIWA PV-AF144V5..... ツクモ特価 ¥49,800
- オムロン MD144XT10V..... ツクモ特価 ¥44,800
- マイクロア MC14400FX..... ツクモ特価 ¥39,800
- Panasonic TO-703B..... ツクモ特価 ¥49,800
- ★通信ソフト
- た〜あのみる2..... ツクモ特価 ¥13,000
- Communication SX-88K ツクモ特価 ¥16,800

冬のボーナス一括払(金利手数料なし)詳しくはお問い合わせ下さい。



ツクモ in 東京 平日 AM10:45~PM7:30  
日・祝 AM10:15~PM7:00  
ツクモパソコン本店II3F

☎03(3253)1899(直通)

担当 ツクモパソコン本店II代表  
荒井 ☎03(3253)4199(毎週木曜日)



ツクモ  
ニューセンター店

☎03(3251)0987

担当 (休) 毎週木曜日  
沢栄



※下取り交換、中古販売も  
行っております。

各店、定休日が祝日と重なる場合は営業致します。

# パソコン・ポケコン・電子手帳・ 関連周辺機器 各種品揃え

## SHARP

新携帯情報ツール  
液晶ペンコム・ZAURUS  
**PI-3000**  
定価¥65,000



## HP

HEWLETT  
PACKARD  
**HP 100LX**  
¥123,000→**特価**  
MS-DOS Ver.5.0/Lotus 1-2-3 R2.4



●アセンブラ・C言語・BASIC・CASLの四言語搭載。業界初、16ビットCPU(インテル8086系)搭載  
●64KバイトRAM標準装備(最大96Kバイトまで拡張可能)

●構造化BASIC命令搭載  
●スクリーンエディタ採用  
●大型FSTN液晶表示  
●その他、新機能搭載

●C言語・BASIC・CASLの三言語搭載  
●32桁×4行  
●大型液晶表示 **在庫処分特価**

●接続ケーブル(CE-T800)で各種パソコンと簡単に接続可能  
●32KバイトRAM

## FX-890P

定価¥34,800→**特価¥24,800**



CASIO

## PC-E650

定価¥33,000→**特価¥26,400**



SHARP

## FX-870P

定価¥28,000→**特価¥18,900**



CASIO

## PC-E200

定価¥22,000→**特価¥17,800**



SHARP

## FX-795P

定価¥19,800→**特価¥15,800**

CASIO

## FX-820P

定価¥29,800→**特価¥19,800**

CASIO

## PC-1262

定価¥24,800→**特価¥19,800**

SHARP

## PC-1280

定価¥24,800→**特価¥21,800**

SHARP

## PC-1360K

定価¥36,800→**特価¥13,500**

SHARP

## PC-1600K

定価¥69,800→**特価¥59,800**

SHARP

## PB-120

定価¥12,800→**特価¥9,800**

CASIO

## CE-515P

定価¥49,800→**特価¥40,300**

SHARP

## MD-120A

定価¥45,000→**特価¥38,200**

CASIO

X1シリーズ				型番 品名				標準価格 特別価格				型番 品名				標準価格 特別価格				型番 品名				標準価格 特別価格			
型番	品名	標準価格	特別価格	MZ-1C17	700用I/Oケーブル	¥2,000		MZ-1R14	MZ-5500用辞書ROM	¥40,000	¥22,000	UE-1R07	AX 辞書ROMボード	¥32,800	¥26,200	●ソフト●				IP-1215	MZ-2500 COBOL		¥11,700	IP-1251	MZ-2800 デスクトップ	¥88,000	¥10,000
CZ-300C	3.5FDDx2(Compact Type)	¥388,000	¥285,000	MZ-1C18	700用I/Oケーブル	¥3,040		MZ-1R16	MZ-5500用増設28KBRAM	¥30,000	¥8,000	UE-1R09	AX IM増設RAMボード	¥75,000	¥55,000	MZ-2500	IBM変換			IP-1253	MZ-2800 クリッパー	¥77,000	¥10,000	IP-1254	MZ-2800 プランニング	¥66,000	¥10,000
CZ-310C	3.5FDDx2 1MBHDD(Compact Type)	¥478,000	¥350,000	MZ-1C24	IP04用プリンタケーブル	¥7,200		MZ-1R21	MZ-IP10第二水準漢字ROM	¥38,000	¥13,000	UE-1R11	AX IM増設RAMボード			MZ-2500	DANGER BOX	¥1,000		IP-1254	MZ-2800 プランニング	¥66,000	¥10,000	MZ-2500	五左衛門	¥1,000	
CZ-500C	5FDDx2 80MBHDD	¥398,000	¥289,000	MZ-1C25	700用プリンタケーブル	¥6,000		MZ-1R24	MZ-1500用辞書ROM	¥22,000	¥6,000	UE-1R13	AX 辞書ROMボード	¥32,800	¥25,000	MZ-2500	ガイダンスコープ	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
CZ-510C	5FDDx2 80MBHDD	¥488,000	¥360,000	MZ-1C26	700用プリンタケーブル	¥6,240		MZ-1R26	MZ-2500用増設RAM		¥10,000	UE-1U01	AX スロットボックス	¥5,000	¥4,000	MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
CZ-820C	X1G MODEL10	¥16,800		MZ-1C32A	5500/6500用プリンタケーブル	¥7,800		MZ-1R35	MZ-2800用1MBRAM		¥19,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
CZ-822C	X1G MODEL30	¥118,000	¥39,800	MZ-1C35	2000/2200用プリンタケーブル	¥6,800		MZ-1S13	MZ-1D17用チャイルドスタンド	¥12,000	¥5,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
CZ-830C	X1 twin	¥99,800	¥35,000	MZ-1E01	MZ-3500用RS-232Cボード	¥28,000		MZ-1T02	MZ-2200用ケーブルコネクタ	¥19,800	¥8,500					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E04	MZ-2000用プリンタケーブル	¥10,000		MZ-1T03	MZ-2500用ケーブルコネクタ	¥12,000	¥8,500					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E08	MZ-2000/2200用プリンタケーブル	¥9,000		MZ-1U04	MZ-2500用拡張ボード	¥278,000	¥75,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E14	MZ-1500用ケーブルコネクタ	¥9,800		MZ-1X22	MZ用モジュールユニット	¥21,800	¥13,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E18	MZ-2000用ケーブルコネクタ	¥9,800		MZ-1X30	MZ用1200/3000モジュール	¥98,000	¥19,800					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E21	MZ-5500用GP I/F	¥36,000		MZ-2Z03	MZ-5500GW-BASIC		¥30,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E22	MZ-5500用GP I/F	¥72,800	¥25,000	MZ-2Z09	MZ-6500TODAY		¥20,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E29	RS-232C I/F 300B	¥17,800	¥9,800	MZ-2Z09	MZ-5500IBM-FORMAT CONVERSION		¥8,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E32	MZ-2500用パラレルI/F	¥30,000	¥27,000	MZ-2Z13	MZ-1500/1600用フォーマット変換ソフト		¥450					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E33	MZ-6500用パラレルI/F	¥34,800	¥28,000	MZ-2Z13	クイックディスク		¥450					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E39	MZ-2800用RS-232C I/F	¥39,800	¥13,000	MZ-6P06	MZ-IP06用トラクタフィード	¥15,000	¥7,500					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E44	MZ-6500用16Bボード	¥50,000	¥15,000	MZ-6P16	MZ-IP22用リカセット(黒)	¥1,500	¥1,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1E45	MZ-6500用RS-232C I/F	¥50,000	¥15,000	MZ-6P17	MZ-IP22用リカセット(カラ)	¥1,500	¥1,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1M01	MZ-2000/2200用16Bボード	¥78,000	¥3,000	MZ-6P20	MZ-IP18/28用リカセット	¥60,000	¥35,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1M03	MZ-5500用数値演算プロセッサ	¥69,000	¥38,500	MZ-6P21	MZ-IP22/17用ロールホルダー	¥3,100	¥2,700					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1M09	MZ-6500用8082-2演算プロセッサ	¥82,000	¥30,000	MZ-6P22	MZ-IP22用リカセット	¥1,800	¥1,600					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1M12	MZ-2800/3000用数値演算プロセッサ	¥90,000	¥45,000	MZ-6P27	MZ-IP22用リカセット	¥58,000	¥39,800					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1P06	ドットプリンタ	¥234,000	¥45,000	MZ-6P29	MZ-IP22用リカセット	¥50,000	¥37,500					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1P10A	24ドット80列漢字プリンタ	¥245,000	¥79,000	MZ-6Z25	MZ-5500用リカセット	¥39,800	¥15,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1P27	熱転写漢字プリンタ	¥59,800	¥16,000	MZ-80P4B	136ドットドットプリンタ		¥48,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1R01	MZ-2000/2200Gボード	¥268,000	¥75,000	MZ-8B10A	MZ-2000/2200用GP I/F	¥45,000	¥18,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1R06	MZ-5500用増設RAM	¥39,800	¥10,000	MZ-8B01	MZ-2000/2200用GP I/F	¥18,000	¥8,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1R09	MZ-5500 VRAM	¥35,000	¥15,000	MZ-8B01	MZ-2000/2200用GP I/F	¥18,000	¥8,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1R10	MZ-5500漢字ROM付	¥30,000	¥9,800	SS-SC28M	MZ-2800/3000用デモソフト	¥49,800	¥10,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1R11	MZ-5500用増設256KBRAM	¥80,000	¥35,000	UE-01	AX ICカードリーダー	¥45,000	¥30,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
				MZ-1R12	MZ-80B/2000/1500/700用RAM	¥35,000	¥8,000	UE-1E02	AX ICカードリーダー	¥45,000	¥30,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	
								UE-1R03	AX IM増設RAMボード	¥100,000	¥65,000					MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000		MZ-2500	ライオン	¥1,000	

(全商品新品完全保証付)  
★シャープ・シャープ周辺機器(拡張機器全機種、プリンタ他)・富士通・NEC取り扱い。  
★シャープ・パナソニック全機種取り扱い。PACIFIC・YHP・キヤノンも取り扱い。  
★上記商品価格には、消費税は含まれておりません。

通信販売のお問い合わせ、御注文は  
**TEL.0426-45-3001(本店) FAX.0426-44-6002**  
●営業時間/10:00～19:00●電話受付/9:00～21:00 迄可●定休日/水曜日  
**SHARP SUPER EXE SHOP**  
アイビット電子株式会社 〒192 東京都八王子市北野町560-5



上記の広告商品は店頭販売もしております。

## 全通販 国信売

北海道から沖縄まで

★送料はご注文の際にお問い合わせ下さい。  
★掲載の商品は、すべて新品、保証書付きです。  
★掲載の商品は充分用意しておりますが、ご注文の際は、在庫の確認の上、現金書留または、銀行振込でお申し込み下さい。全商品クレジットでも扱っております。  
★お申し込みの際は必ず電話番号を明記して下さい。  
★商品、品切れの際はご容赦下さい。

富士銀行八王子支店 (普)1752505



## SHARPパーソナルワークステーションX68000用サブMPUボード **POLYPHON (ポリフォン)**

POLYPHONの供給クロックが16MHzから24MHzになりました。速度比でノーマルPOLYPHONの約1.5倍、X68000本体の約2.4倍です。これなら時間のかかったコンパイルもX68030に買い換えることなく解消されるでしょう。

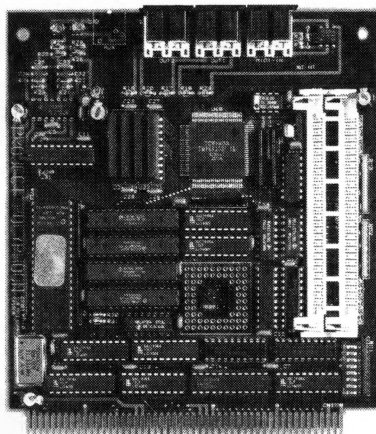
### POLYPHON標準価格

POLYPHON 8Mモデル	¥85,000
POLYPHON 8Mモデル(68881付)	¥95,000
POLYPHON 0Mモデル	¥62,000
POLYPHON 0Mモデル(68881付)	¥72,000

POLYPHON-24の出荷は12月以降のロット分からとなっております。それ以前にお買い求めになられたユーザーの方のために、クロックモジュールアップグレードを用意しております。近日、購入ユーザーの方には案内状を送付いたしますので、今しばらくお待ちください。

### システムディスク Ver.Up 受付中

POLYPHONシステムディスクのバージョンアップを受け付けています。随時最新の内容でお届けします。ご希望のユーザーは62円切手6枚を希望メディアを明記した上で、弊社まで送ってください。(ブランクディスク2枚と返送用切手でも可)



### 購入方法

弊社製品は直販のみの販売でSHOPではお求めになれません。詳しい購入方法や細かい仕様などの資料を用意しておりますので、郵便番号・住所・氏名を明記の上、ハガキにてご請求ください。難しい文字には読み仮名を付けて下さい(代金を直接送らないようお願いします)。電話でのお問い合わせも受け付けておりますが、業務の都合留守電に繋がる事もありますのでご了承下さい。

### ソフト・ハード投稿受付中

あなたの作品を製品化致します。ハード・ソフト問いません。我こそはという方は是非とも投稿してみてください。特にソフトウェアは歓迎致します(即、製品化も夢ではありません)。左記の住所まであなたの作品の使用法や詳しい資料などと一緒に送ってください。審査・検討の上、製品化の場合は規定のロイヤリティを支払い致します。

### 新製品近日登場

POLYPHONに続く製品を期待される声を生かし、日夜開発を続けています。  
XVI用内蔵メモリ 94年2月発売予定・価格未定  
拡張I/O BOX 発売日・価格未定  
上記製品は開発中のため、詳細や発売日などのお問い合わせはご遠慮ください。

### X680x0用外付大容量ハードディスク

プログラム・音楽データ・画像データ...とハードディスクの足りない方にオススメ。フォーマット済のため、接続後にすぐ使用できます(パーティション分割する場合は、一旦領域解放し、再度領域を確保してください)。

1.0GB (Quantum社製ドライブ使用)	¥168,000-
1.2GB (Quantum社製ドライブ使用)	¥198,000-
2.4GB (Seagate社製ドライブ使用)	¥348,000-

すべてケーブル付。

その他の容量も取り扱っていますので、お問い合わせください。

### サポートネットのご案内

POLYPHONやPCM8 (SB) などのサポートはネットワークでも御利用いただけます。最新の情報やプログラムが入手できます。9/25よりISDNにも対応したため、最高で38400bpsでの通信が可能となりました。

#### 回線番号

03-5680-7533	300~14400bps
03-5680-7534	300~9600bps, 9600~38400bps (INS-C)
ゲストID	Guset (パスワードは必要ありません)

### 株式会社ネオコンピュータシステム

120 東京都足立区綾瀬1-33-7-103

TEL 03-5680-7531 (Mon-Fri AM10:00-PM4:00)

FAX 03-5680-6810 (24hours)

NET 03-5680-7533, 03-5680-7534 (24hours)

お詫び：プログラム制作遅延の為発売を延期します。発売目標12月上旬...

GOMENNASAI!

# Mu-1 GS

■ ローランド社 SC-55mk-II SOUND Canvas対応 / MIDIマルチレコーダー

- ☆使いやすくなったGS音源エディット※
- ☆RS-232C/MIDI出力対応  
(注意:出力のみ対応、単独使用不可/要MIDIボード)
- ☆簡単エクスクループ入力
- ☆シーケンス機能はMu-1 Super (X68030/25MHz対応)
- ☆スタンダードMIDIファイル対応
- ☆ミュージング II データコンバート機能追加
- ☆国本佳宏/GS対応デモ曲収録

※Mu-1, Mu-1 Superのユーザーの方々には、バージョンアップのご案内をお送りいたします。

※下記コントロールコードのリアルタイムエディットおよび任意の位置へのステップ入力が可能  
 ◎リバーブ、コーラスのセンドレベル、パンポット、バンク、レベル、キープ  
 ◎TVFカットオフ周波数、TVFレゾナンス、TVF8TVA、アタック、ディケイ、リリース  
 ◎ドラムインストルメント、ビッチ、リバーブセンド、コーラスセンド、パンポット、ボリューム

通信販売の方法：現金書留にて右記の宛先  
 “Mu-1通販係”まで代金をお送りください。  
 必ず、住所、氏名、電話番号を記入してください。

音質重視！内蔵FM音源とMIDI音源のオーディオアウトをミックスする  
**オーディオ拡張キット (システムサコム社製SX68M II用)**

※スロットカバーは黒のみ

通販のみ/¥8,000(送料・税込)

標準価格 ¥28,000 (税抜き)

### Mu-1 Superキャンペーン版

GSお試し版同梱 / ¥25,000 (税抜)

### キャンペーン特価/発売中

通販のみ/MIDIボード付 ¥30,000 (送料・税込)

※登録ユーザーの方々にはMu-1 GSへの有償バージョンアップ (¥3,000) のご案内をお送りします。

〒213 神奈川県川崎市高津区下作延1043

株式会社 サンワード

TEL 044-855-4335

for **△X680x0 Series Only**  
**オリジナル アプリケーション**  
 開発速報#3

R&D Division  
 of  
 計測技研  
*FirstClassTechnology*

新発売

# CD-ROM Driver Ver1.06

定価  
**¥4,800**

## まだ、フロッピーディスクですか？

「X680x0専用CD-ROM」はまだまだ少ないけれど(すみません頑張ります)、X680x0にCD-ROMを接続することで、あなたの創造の世界はぐっと広がります。

CD-ROMはハードディスクやMOの代わりにはなりませんが、大量のデータを、安価に、たくさんの人に配布するためのメディアとしては、いまのところこれ以上のものはありません。すでにMacやAT互換機、そしてワークステーションの世界では、CD-ROMによって大量のデータが日常的にやりとりされているのです。

その中にはX680x0で利用できるデータが山ほど存在します。  
 たとえば...

### ★グラフィッカー向け

- ・Mac/AT互換機/Amigaの画像データ ※1※2
- ・PhotoCD

### ★デスクトップミュージシャン向け

- ・標準MIDIファイル※1
- ・オーディオCDもちろんOK

### ★プログラマー向け

- ・Mac/AT互換機/Amiga/ワークステーションのソース、技術資料

### ★ライター向け

- ・各種辞書タイトル → **続報をお待ちください**

CD-ROM Driver Ver1.06は、Human68k上でCD-ROMをフロッピー感覚で扱えるようにするデバイスドライバです。

あなたも世界を結ぶデータの大海に乗り出してみませんか？

CD-ROM Driver Ver 1.06は以下のCD-ROMドライブに対応しています。

東芝製ドライブ(KGU-XCD, KGU-XCD II)  
 ソニー製ドライブ  
 パイオニア製ドライブ  
 NEC製ドライブ  
 (他社製ドライブも確認中)

### X680x0用フリーソフトウェア集CD-ROM FreeSoftwareSelection Vol.1

(定価¥5,000)も好評発売中です。

※1 ISO9660フォーマットのメディア、またはMacintosh HFSフォーマットの場合

※2 X680x0で扱える画像フォーマットの場合

※ 記載されている会社名および商品名は各社の登録商標もしくは商標です。

発売中

## SX-PhotoGallery

「Kodakフォトサンプラー」CD

バンドルセット

¥19,800

基本セット

¥15,800

PhotoCDのフルカラー記録を、SX-WINDOW Ver.3.0のグラフィックウィンドウで美しく再現します。

SX-WINDOWの特長である、カット&ペーストによるアプリケーション間でのデータのやりとりにも対応。また、PhotoCDの画像展開モジュールはIVM.X用のリソースとして用意しましたので、キャンパス、シャープペン、Easydraw、EasypaintなどでPhotoCD画像を利用することができます。

SX-PhotoGalleryにはCD-Driverが付属します。

東芝製ドライブ(または同等品)、またはCD-ROM XA自動対応ドライブが必要です。

開発中

## スケジューラソフト for SX-WINDOW(名称未定)



本誌9月号で第一報をお知らせしたスケジューラソフトも11月発売を目標に好調開発中です。

ご期待ください。

画面は開発中バージョンです

お求めはお近くのパソコンショップ、または弊社通販部(TEL:0286-22-9811)へお申し込みください。

低金利クレジット 通信販売送料 全国一律¥1,000 長期クレジット可能

株式会社 計測技研

マイコンショップ

BASIC HOUSE

本社 / ショールーム / 通販部

※表示価格に消費税は含まれておりません

〒321 栃木県宇都宮市竹林町503-1

TEL 0286-22-9811

FAX 0286-25-3970



パソコン/ワープロ通信ネットワークサービス  
**J&P HOT LINE**



# ネットワーカー・ネットワーク



## 第10回 スッパマンさん ID: JH025935

今回登場されるのは、SHARPのXシリーズマシンの大ファンと自認されているスッパマンさん。Oh/Xも創刊号の頃からご愛読いただき、J&P HOT LINE内でも、SIG (CZ-CLUB) のハードウェア関連のサブオベとして大活躍されています。そんなスッパマンさんにエッセイの形でXシリーズとJ&P HOT LINEについて語っていただきました。

J&P HOT LINEには、実験開局の頃からのおつきあいです。最近、SIGの、CZ-CLUB、SHARP-HOTLINE、おさがせ村サリーに出入りしてます。

CZ-CLUBでは、回覧ディスクの世話人をしています。これはXシリーズユーザーにとって市販ソフトを補完するものとして比重の大きいフリーウェアの入手と、ダウンロードする通信費の軽減、それに個人で製作したソフトやPDDなどの貴重なデータの交換を行うことを目的として提案されたものです。OL Tの中で話が決まりましたが、ただ、あまりの量に、なかば本気で、「MO回覧にしようよ」という声もあります。みなさんもぜひ参加してください。

今の環境は、最古参X1 turboが現役機として活躍しています。通信端末は8ビットで十分というのが私の考えです。データ処理の都合上、どうしても必要な時はX68000を使います。ただ最近X68030も欲しいな……と思っています。

休日は、主に日本橋を歩いたり、ハード工作をしたりしています。X68の外付け3.5FDDや、総費用1万5千円でジャンク品の75M-

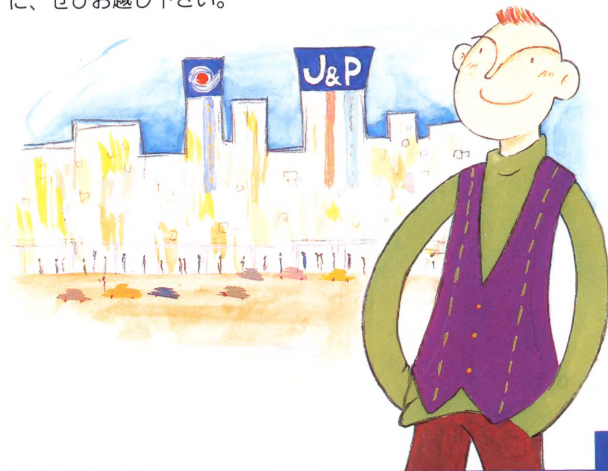
### =基本データ=

所有機種名: X1 (元祖) が2台、X1Dが1台  
X1 turbo M30が2台  
X1 turbo Z/ZIIが各1台  
X68000 (元祖) MZ-2521が1台  
周辺その他: HDD (75Mが2台+40M)……X68  
CM-32L MZ-1X30、CZ-8TM2、  
MD2400F その他多数

HDDを活用したり、古い嬉楽画マウスを改造した98マウス用のアダプタ (Oh/Xの記事より前に作ったのはささやかな自慢です) を利用しています。

これらのハード情報の交換のために、CZ-CLUB内に「周辺機器研究室」を設けています。

自作派としてひとこと言わせていただくと、COMPACT以降の拡張FDDやイメージ入力端子のコネクタは入手困難という事でみなさん苦勞しているようです。増設FDD等は自作を考える人は多いと思いますが……。そんなハードの話もできるCZ-CLUBに、ぜひお越し下さい。



J&P HOT LINEへのご入会はスタータキットで。

買ったその日から  
2週間無料で  
アクセスできます。

お求めは、下記のお店へ。又は現金書留にて、¥3,000+¥90(消費税3%)=¥3,090を事務局までお送り下さい。すぐにスタータキットをお送りします。

お問い合わせは  
〒556 大阪市浪速区日本橋西1-6-5 上新電機株式会社  
J&P HOTLINE事務局宛 TEL.(06) 632-2521

### スタータキットのお求めはJ&P各店でどうぞ。

渋谷店 ☎(03)3496-4141  
町田店 ☎(0427)23-1313  
八王子店 ☎(0426)26-4141  
立川店 ☎(0425)36-4141  
三鷹店 ☎(0422)31-6251  
本厚木店 ☎(0462)25-5151  
津田沼店 ☎(0474)72-5211

越谷店 ☎(0489)66-1221  
焼津インター店 ☎(054)626-3311  
にいがた1ばん館 ☎(025)241-3711  
富山店 ☎(0764)22-5033  
金沢店 ☎(0762)91-1130  
寺地店 ☎(0762)47-2524  
熊本店 ☎(096)359-7800

大須店 ☎(052)262-1141  
テクノランド ☎(06) 634-1211  
メディアランド ☎(06) 634-1511  
コスモランド ☎(06) 634-3111  
U.S.LAND ☎(06) 634-1411  
ビジネスランド ☎(06) 348-1881  
高槻店 ☎(0726)85-1212

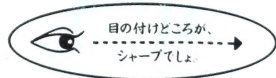
くすは店 ☎(0720)56-8181  
千里中央店 ☎(06) 834-4141  
摂津富田店 ☎(0726)93-7521  
寝屋川店 ☎(0720)34-1166  
枚方バイパス店 ☎(0720)48-1211  
藤井寺店 ☎(0729)38-2111  
岸和田店 ☎(0724)37-1021

さんのみやばん館 ☎(078)231-2111  
西宮店 ☎(0798)71-1171  
伊丹店 ☎(0727)77-5101  
姫路店 ☎(0792)22-1221  
京都寺町店 ☎(075)341-4411  
京都近鉄店 ☎(075)341-5769  
大久保バイパス店 ☎(0774)44-1211

和歌山店 ☎(0734)28-1441  
和歌山南店 ☎(0734)25-1414  
学園前店 ☎(0742)49-1411  
奈良1ばん館 ☎(0742)27-1111  
新大宮店 ☎(0742)35-2611  
郡山インター店 ☎(07435)9-2221  
田原本店 ☎(07443)3-4041



# SHARP



## **68030** 32bit PERSONAL WORKSTATION

ピュア32bit MC68EC030搭載。  
クリエイティブパワーが花開くX68030シリーズ。



### **X68030**

本体+キーボード+マウス+トラックボール  
5.25インチFDDタイプ CZ-500C-B(チタンブラック)標準価格398,000円(税別)  
HDタイプ CZ-510C-B(チタンブラック)標準価格488,000円(税別)

**NEW**

### **X68030 Compact**

本体+キーボード+マウス  
3.5インチFDDタイプ2DD対応 CZ-300C-B(チタンブラック)標準価格388,000円(税別)  
HDタイプ CZ-310C-B(チタンブラック)標準価格478,000円(税別)



●写真のカラーディスプレイは別売です。

## なか身は、どちらも32ビット。

プロセッサの未来を先取、洗練されたアーキテクチャを誇るMPU MC68000シリーズを搭載。  
先駆のクリエイティブ・アビリティで使う人の創造性に応える68ワールドへ、どうぞ。

## **68000** PERSONAL WORKSTATION・XVI

32bit内部演算処理、16bitバスアーキテクチャ。  
潜在能力を秘めたX68000シリーズ。



### **X68000 XVI**

本体+キーボード+マウス+トラックボール  
5.25インチFDDタイプ CZ-634C-TN(チタンブラック)標準価格368,000円(税別)

### **X68000 XVI Compact**

本体+キーボード+マウス  
3.5インチFDDタイプ CZ-674C-H(グレー)標準価格298,000円(税別)



※X68000シリーズはMC68000(内部レジスタ32ビット、16ビットバス)を搭載しています。●写真のカラーディスプレイおよびカラーディスプレイテレビは別売です。

●お問い合わせは...

シャープ株式会社 コンシューマセンター西日本相談室〒545大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表) 電子機器事業本部システム機器営業部〒545大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表)



T1002179110602 雑誌 02179-11